NPS-54-82-003A

# NAVAL POSTGRADUATE SCHOOL

Monterey, California

FUNCTIONAL DESIGN OF A LOCAL AREA NETWORK FOR THE STOCK POINT LOGISTICS INTEGRATED COMMUNICATIONS ENVIRONMENT

Norman F. Schneidewind

December 1982

Final Report: 15 Mar 82 to 1 Dec 82

Prepared for:
Fleet Material Support Office
Mechanicsburg, Pennsylvania

NAVAL POSTGRADUATE SCHOOL
Monterey, California

The work reported herein was supported by the Fleet Material Support Office.

This report was prepared by:

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

| | |
|---|---|
| REPORT DOCUMENTATION PAGE | READ INSTRUCTIONS BEFORE COMPLETING FORM |
| 1. REPORT NUMBER NPS-54-82-003A / 2. GOVT ACCESSION NO. | 3. RECIPIENT'S CATALOG NUMBER |
| 4. TITLE (and Subtitle) FUNCTIONAL DESIGN OF A LOCAL AREA NETWORK FOR THE STOCK POINT LOGISTICS INTEGRATED COMMUNICATIONS ENVIRONMENT | 5. TYPE OF REPORT & PERIOD COVERED Final Report 15 MAR 82 - 1 DEC 82 |
| | 6. PERFORMING ORG. REPORT NUMBER |
| 7. AUTHOR(s) Norman F. Schneidewind | 8. CONTRACT OR GRANT NUMBER(s) |
| 9. PERFORMING ORGANIZATION NAME AND ADDRESS Fleet Material Support Office Mechanicsburg, Pennsylvania | 10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS N0036782PON3907 |
| 11. CONTROLLING OFFICE NAME AND ADDRESS Fleet Material Support Office Mechanicsburg, Pennsylvania | 12. REPORT DATE December 1982 |
| | 13. NUMBER OF PAGES 73 |
| 14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) | 15. SECURITY CLASS. (of this report) Unclassified |
| | 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE |

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

| | |
|---|---|
| Data Base Management | Local Area Network |
| Defense Data Network | Session Services |
| Communication Protocols | SPLICE |
| Functional Modules | Terminal Management |

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

As a result of the growing demands for automated data processing at the Navy stock points and inventory control points, long range plans are being developed around the Stock Point Logistics Integrated Communications Environment (SPLICE) concept. This report provides a design and implementation strategy which is based on a distributed architecture for a Local Area Network (LAN).

The features of our functional design and the points which we recommend

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

for adoption by the Fleet Material Support Office and the Naval Supply Systems Command are the following:

° Generalized functional modules (e.g., Terminal Management) for serving all applications.

° Virtual LAN design approach which places emphasis on the functional requirements of the LAN and later maps to the best physical implementation.

° Communications protocol within each LAN which only contains the complexity necessary to support intra LAN communication and which interfaces to the Transmission Control Protocol and Internet Protocol for communication over the Defense Data Network.

° Virtual bus structure for communication between functional modules and distributed, as opposed to centralized, system control.

° Flexible screen and report format management achieved through user-provided screen and report designs.

° Virtual terminal protocol for achieving independence from specific terminal characteristics.

° Geographically distributed data base management for inter-LAN transactions but centralized data base management for intra LAN <u>interactive</u> transactions, using the concept of a specialized file server module, with a form of distribution existing on each LAN due to the separation of batch oriented main frame dbms from interactive dbms operating in SPLICE minicomputers.

## TABLE OF CONTENTS

## LIST OF FIGURES

## LIST OF TABLES

## LIST OF ABBREVIATIONS AND ACRONYMNS

| | |
|---|---|
| DBM | DATABASE MANAGEMENT |
| DDN | DEFENSE DATA NETWORK |
| FEP | FRONT END PROCESSOR |
| FM | FUNCTIONAL MODULE |
| IMP | INTERFACE MESSAGE PROCESSOR |
| IP | INTERNET PROTOCOL |
| LAN | LOCAL AREA NETWORK |
| LC | LOCAL COMMUNICATIONS |
| NC | NATIONAL COMMUNICATIONS |
| PSN | PACKET SWITCHING NODE (EQUIVALENT TO AN IMP) |
| RA | RESOURCE ALLOCATION |
| RM | RECOVERY MANAGEMENT |
| RST | RESOURCES STATUS TABLE |
| SPLICE | STOCK POINT LOGISTICS INTEGRATED COMMUNICATIONS ENVIRONMENT |
| SS | SESSION SERVICES |
| TCP | TRANSMISSION CONTROL PROTOCOL |
| TM | TERMINAL MANAGEMENT |

## SUMMARY

The purpose of this report is to provide a functional design specification for a SPLICE Local Area Network which can be used as a baseline for developing the detailed system design. The features of this design, and the points which we recommend for FMSO and NAVSUP adoption in SPLICE, are the following:

° Generalized functional modules (e.g., Terminal Management) for serving all applications vice designing and programming individual applications.

° Related to the above point, is the differentiation among applications at the human-machine interface, i.e., at the terminal display and hard copy output.

° Virtual LAN design approach which places emphasis on the functional requirements of the LAN and later maps to the best physical implementation for achieving the functional requirements.

° Communications protocol within each LAN which only contains the complexity necessary to support intra LAN communication and which interfaces to the Transmission Control Protocol and the Internet Protocol for inter LAN communication, i.e., over the Defense Data Network.

° Virtual bus structure for communication between functional modules and distributed, as opposed to centralized, system control.

° Flexible screen and report format management achieved through user-provided screen and report designs.

° Virtual terminal protocol for achieving independence from specific terminal characteristics.

- Geographically distributed data base management for inter-LAN transactions but centralized data base management for intra LAN <u>interactive</u> transactions, using the concept of a specialized file server module as opposed to a data base machine. There will be a form of distribution existing on each LAN due to the separation of batch oriented main frame dbms from interactive dbms operating in SPLICE minicomputers.

## INTRODUCTION

As a result of the growing demands for automated data processing at the Navy stock points and inventory control points, long range plans are being developed around the Stock Point Logistics Integrated Communications Environment (SPLICE) concept. This report provides a design and implementation strategy which is based on a distributed architecture for a Local Area Network (LAN). The feasibility of a distributed LAN has been shown in various applications [1].

SPLICE is designed to augment the existing Navy stock point and inventory control point ADP facilities which support the Uniform Automated Data Processing System - Stock Points (UADPS-SP). The hardware for the UADPS-SP consists of the Burroughs medium size (B-3500/3700/4700/4800) systems. At present there are twenty new application systems being developed which require considerable interactive and telecommunication support. The current UADPS-SP cannot support these requirements without a total redesign effort and will probably require future replacement of the current mainframes. At present, project managers are developing the new application systems utilizing a variety of minicomputers which are capable of supporting the required interactive and telecommunication capabilities. This is being done due to the near term needs of the Navy and are scheduled to be implemented within the next four to five years. There are two major objectives behind the development of SPLICE. First, there is the increased need for the use of CRT display terminals to interact with application logic and to fetch information from the system data base. Second, there is the need to standardize the multitude of interfaces currently existing across approximately sixty supply sites. Standardization of processors is desired to help reduce the overall costs of the SPLICE system with regard to processor and telecommunication support. The SPLICE processors will be co-located with the host Burroughs system at each Navy Stock Point (SP) and with the Burroughs and Univac systems at the Inventory Control Points (ICP's). SPLICE is to provide economical and responsive support capabilities for a distributed telecommunication environment. A "foreground/background" concept will be implemented with SPLICE minicomputers, which will serve as a front-end-processor for the Burroughs systems via a Local Area Network (LAN) interface. The Burroughs computer will provide background processing functions for large file processing and report generation. SPLICE will be developed using a standard set of minicomputer hardware and software. This standardization is particularly important when considering the fact that SPLICE will be implemented at some sixty different geographical locations, each having a different mix of application and terminal requirements. Additionally, each LAN must have the capability of communicating with other LANs via the Defense Data Network (DDN), which is to be provided by the Defense Communications Agency.

## DESIGN APPROACH

We take the approach of designing the logical or virtual Local Area Network (LAN) first, specifying all the functional modules, their characteristics and the communication protocols, rather than focusing on the hardware characteristics of LAN first. This is done so that we can ensure, within the hardware and software constraints of commercially available networks, that functional requirements are satisfied. In a later phase of this project the virtual LAN requirements will be mapped onto a physical local network. The reverse process - starting with a physically defined local network (e.g., CSMA/CD) and working toward the virtual LAN - could result in a dysfunctional LAN. The risk, of course, in our approach is that, after defining the logical LAN, we may find no existing physical LAN which suffices. If this were the case, it would indicate that the available physical LANs are infeasible for the SPLICE application. It is not anticipated that this will be the case.

## FUNCTIONAL MODULES

A functional module is one which provides a generalized function for many applications (e.g., terminal management, data base management). Rather than n sets of application modules, there will be one set of functional modules which will provide services for n applications. We consider this concept a key idea in our design approach and one which would save FMSO considerable time and money in system development and implementation. The traditional approach to system development involves designing and implementing application modules for each application. This is a wasteful approach because many functions (e.g., edit, data base management, report generation, etc.) are common to many applications, resulting in a great deal of redundant system analysis, design, programming, coding, testing and maintenance. This approach also causes a significant increase in the use of computer resources - memory and file storage space. It is primarily the input and output formats and application parameters (e.g., time of printing a periodic report) which differ among applications and not the basic operations of editing data, maintaining files and generating reports and displays. Figures 1 and 2 show the traditional and generalized approaches, respectively. The generalized approach is emphasized in the design of the LAN.

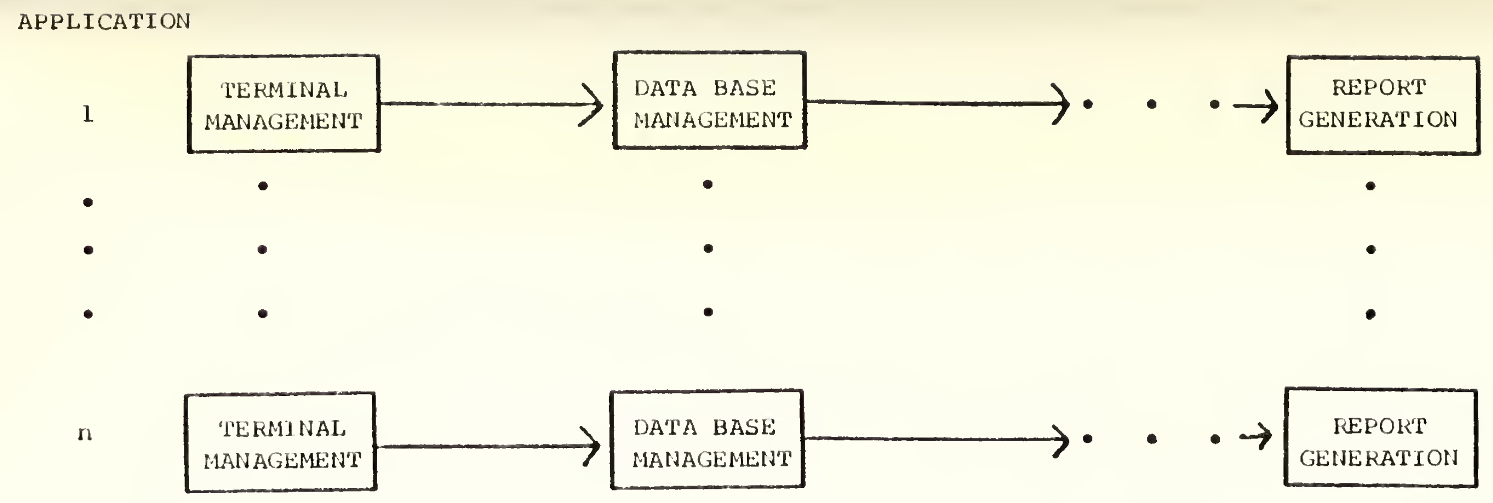


Figure 1 . . . Traditional Approach to System Development

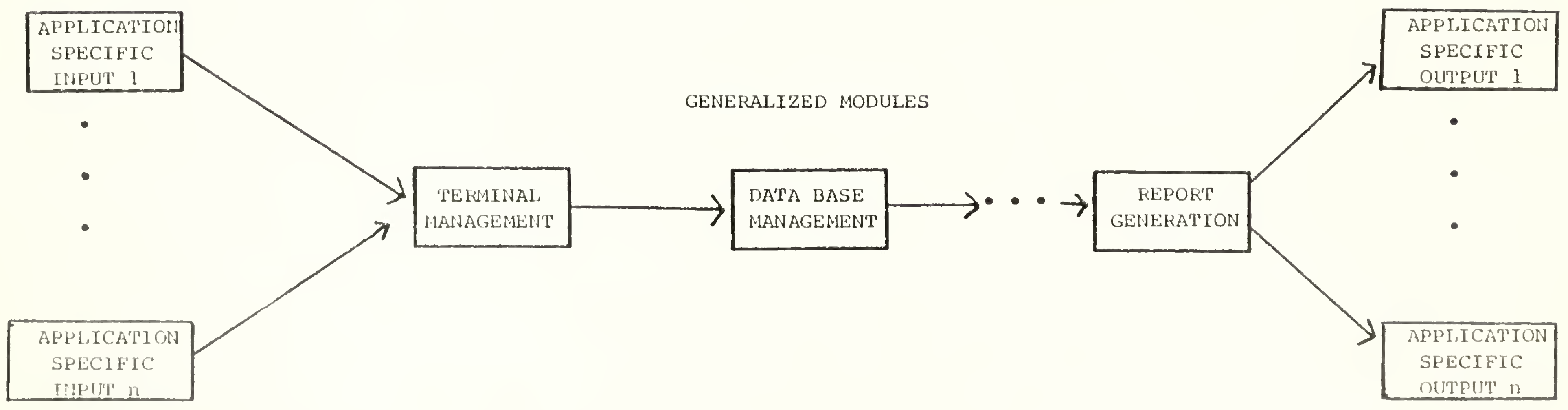

APPLICATION SPECIFIC INPUT 1
APPLICATION SPECIFIC INPUT n
GENERALIZED MODULES
TERMINAL MANAGEMENT
DATA BASE MANAGEMENT
REPORT GENERATION
APPLICATION SPECIFIC OUTPUT 1
APPLICATION SPECIFIC OUTPUT n

Figure 2 . . . Generalized Approach to System Development

## OVERVIEW OF FUNCTIONAL SOFTWARE MODULES AND DESIGN OF VIRTUAL LOCAL AREA NETWORK

In accordance with the previous two sections in which the ideas of virtual LAN design and use of functional modules were mentioned, this section provides an overview of the services provided by various functional modules (Table 1) and the method of providing communication among these modules (Figures 3-5). A bus oriented architecture, as shown in Figures 3-5, has been successfully employed in many LANs [2,3]. There are also more commercially available implementations of this architecture than there are for the competing ring and star configurations.

TABLE 1

Functions of Software Modules

1. Local Communications (LC)
   - ∘ Bus arbitration, i.e., traffic management
   - ∘ Message transmission and reception including buffer management
   - ∘ Message control (e.g., error detection, correction and acknowledgement)
   - ∘ Administration
     - Message accounting
     - Lost or misdirected message handling
     - LAN recovery and shutdown
2. National Communications (NC)
   - ∘ Conversion of Defense Data Network protocol to LAN protocol and vice versa
   - ∘ Message assembly/disassembly
3. Front-End Processing (FEP)
   - ∘ Terminal and communication line buffering
   - ∘ Code conversion
   - ∘ Byte/word assembly/disassembly
4. Terminal Management (TM)
   - ∘ Message editing
   - ∘ Screen management
   - ∘ Virtual terminal operations
5. Data Base Management (DBM)
   - ∘ File creation
   - ∘ File update
   - ∘ Query processing and data retrieval

° Data dictionary creation and maintenance

° File catalog creation and maintenance

6. <u>Session Services</u> (SS)

   ° Establish and maintain local and remote sessions:

     - Within the LAN (SPLICE minicomputer processes)

     - With local host(s) (mainframe processes)

     - With remote host(s) (mainframe processes)

   ° Provide logical and physical network addresses based on value of Services Request Code.

7. <u>Peripheral Management</u> (PM)

   ° Management of Unit Record Input/Output

     - Read a card

     - Print a line, etc.

     - Spool files for input and output

   ° Optical Character Recognition or Mark Sense Equipment

8. <u>Resource Allocation</u> (RA)

   ° Allocation of shared resources to functional modules

     - Record keeping concerning allocation of shared resources

     - Locating, accessing and making shared resources (e.g., memory, disk) available to functional modules

With regard to TABLE 1, items which warrant elaboration are the following:

° Protocol conversion is provided in the National Communications Module because the LAN does not need, and would be slowed down by, the many layers of protocol which are required on the DDN.

° Virtual terminal operation refers to the capability of using a variety of terminals in SPLICE and converting these terminal protocols to a standard Network Virtual Terminal (NVT) protocol for message transfer on the LANs and DDN, with conversion from the NVT protocol to the protocol of the remote terminal or host. This feature is included on the assumption that a variety of terminals will be used in SPLICE and that it would not be possible to standardize on terminal hardware.

- Data Base Management includes a data base management package which would provide languages for file creation, maintenance and query in support of all SPLICE applications, consistent with the generalized approach to system development shown in Figure 2.
- Session Services provides for a session to be established by a terminal user with any process in SPLICE, whether it be a local or remote process, within the constraint of access authority.
- Peripheral Management includes provision for optical character recognition or mark sense equipment in anticipation of possible use of source entry forms (e.g., stock requisition form) as an alternative to masses of supply clerks keying transactions into terminals. This alternative is related to the idea that most users will want to have hard-copy of their requisitions; therefore, when the requisition is typed, it could be typed in a format and font acceptable to OCR computer input.

Items which should be highlighted in Figure 3 are the following:

- Modules are divided into two main categories: operating functions, i.e., transaction processing modules and support modules, i.e., those that exist to make effective use of the processing modules and the entire LAN.

Conceptually, the logical design provides two types of busses: a data bus for transferring the actual application messages and a control bus for carrying administrative traffic (e.g., resource allocation and error messages). Although a physical design of this type would be highly desirable from user and system effectiveness standpoints, because it would reduce congestion of user data messages, few local networks available from vendors provide separate data and control busses because of the additional hardware and cost.

- "Resource Status Table" refers to tables where shared resources availability information (e.g., memory, disk) will be stored. As opposed to Figure 3, which shows the logical network concept, Figure 4 shows a typical three minicomputer LAN physical configuration. Host computers refers to existing and future main frames at SPLICE sites. Terminals would be interfaced to the FEP from both local (e.g., NSC Oakland) sources and satellite (e.g., NARF Alameda) sources. Associated with the NC module is the Transmission Control Protocol (TCP), and the Internet Protocol (IP), the DOD standard transport protocol and network protocols, respectively.
- Figure 5 shows that a user task could involve message flow between a terminal and a locally connected functional module or between a terminal and a remotely connected (e.g., ICP) module. In the latter case, an outgoing message (I2) would go through the NC module for conversion to the format required on the DDN. Conversely, an incoming message (O2) would have its format converted to an LAN compatible pattern before it is displayed to the terminal user. In both cases, messages undergo processing by the FEP for buffering and message assembly/disassembly and by the TM module in order to provide presentation services to the user. In the case of a message internal to a LAN (I1, O1), the message does not flow through the NC module. However, it does undergo processing in the FEP and TM modules in the same manner as for DDN messages.

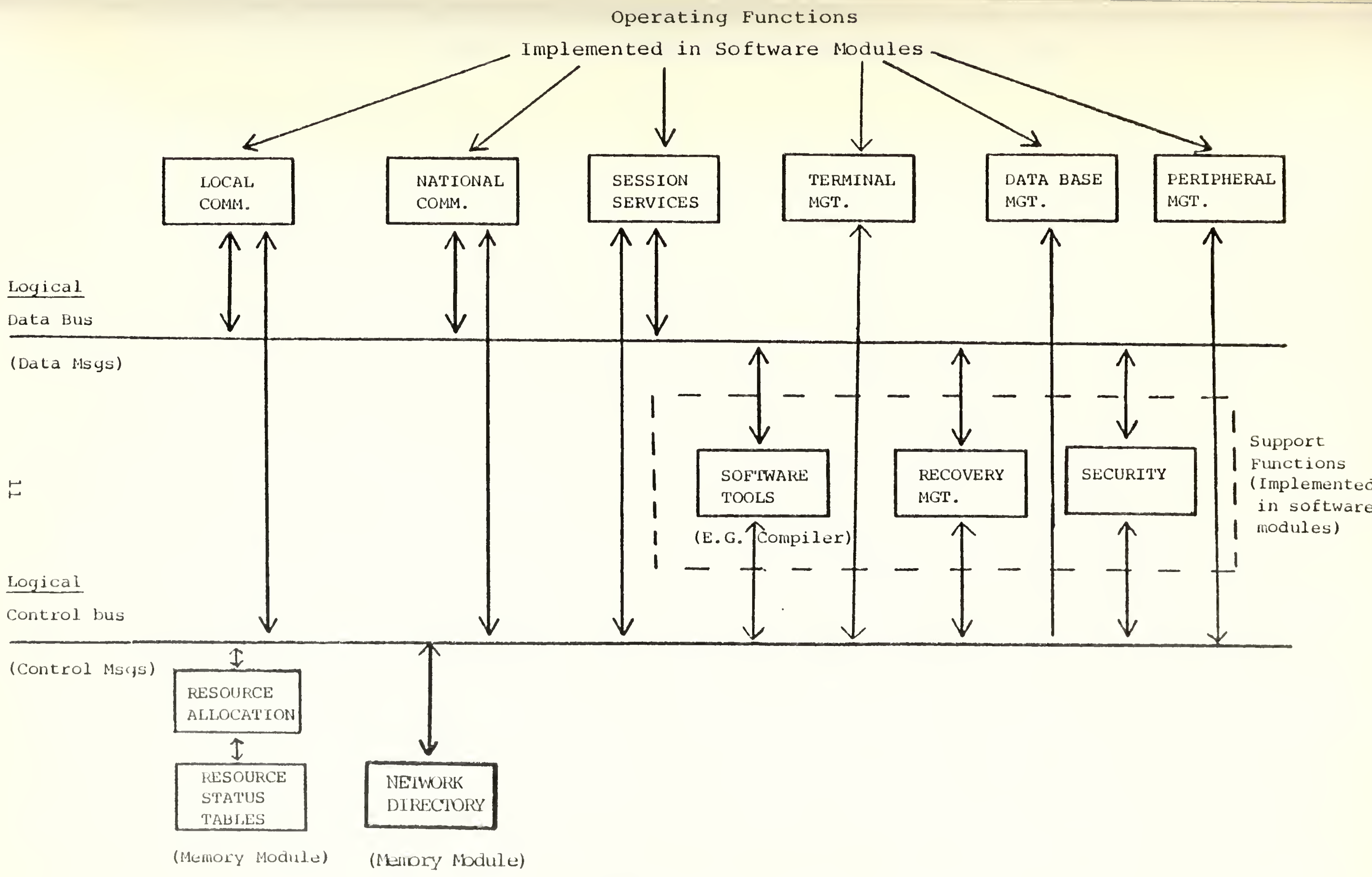
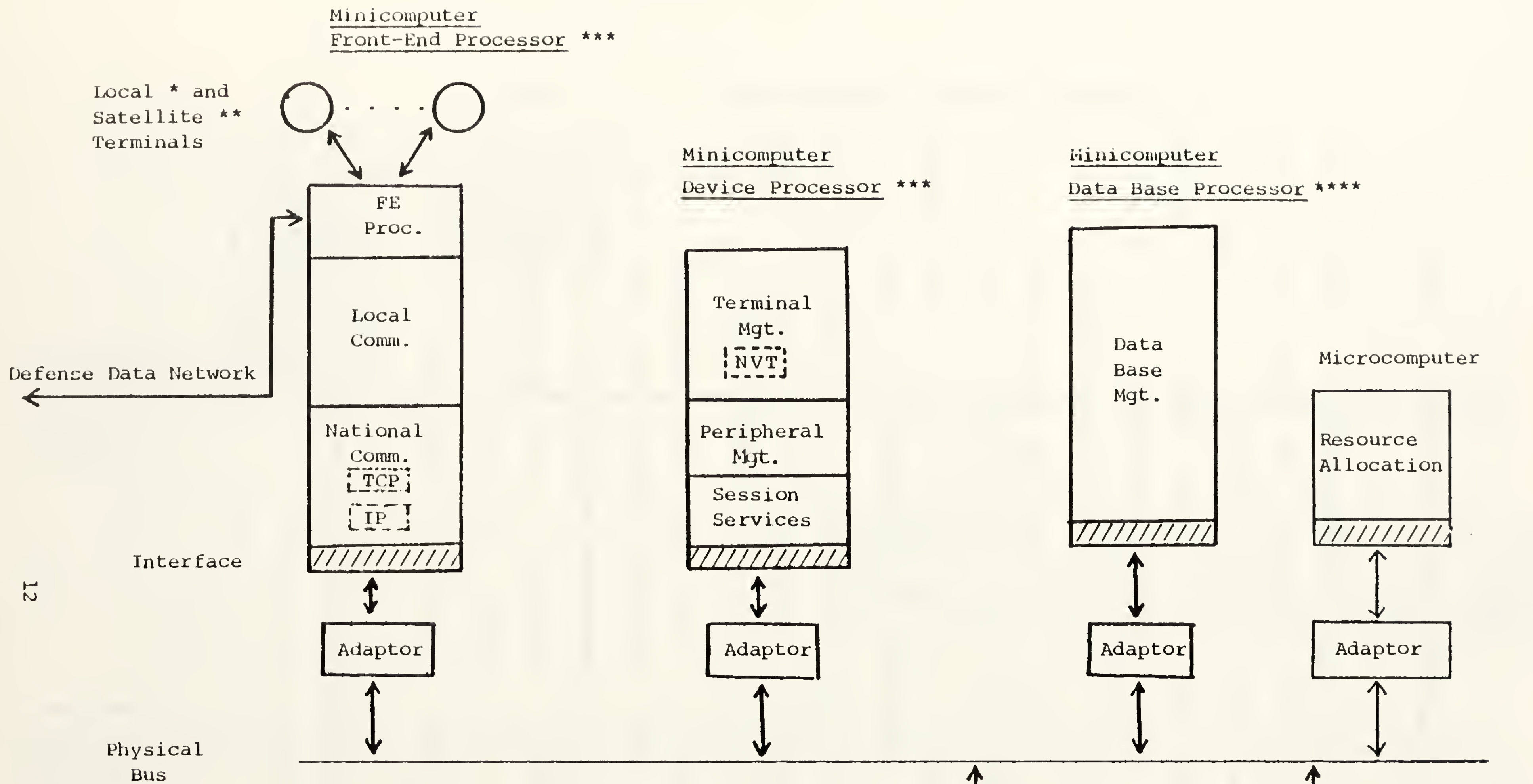


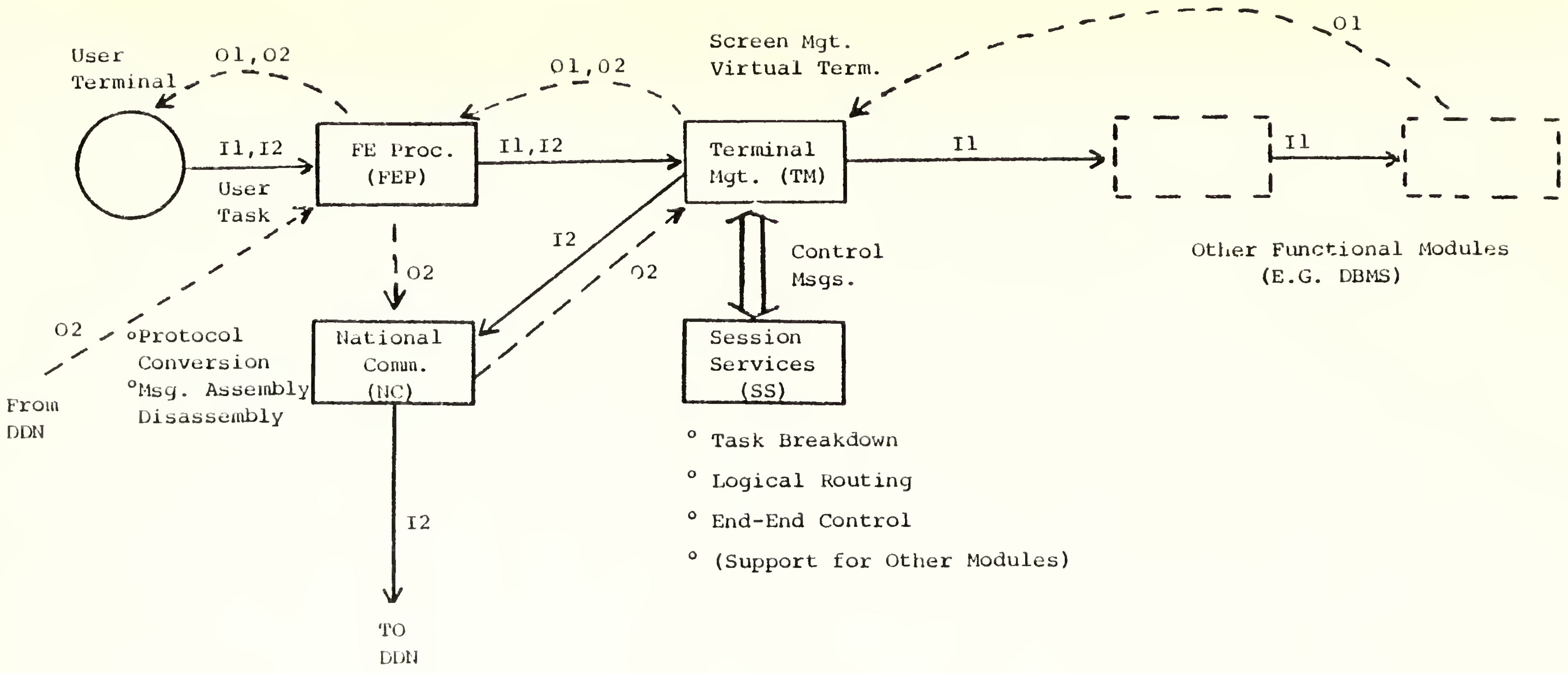
Figure 3 . . . Local Network Logical Connections



TCP: Transmission Control Protocol
IP: Internet Protocol
NVT: Network Virtual Terminal

* SP, ICP
** (E.G. NARF)
*** Dedicated Resources (E:G. Memory, disks associated with each processor)
**** Could be multiple processors

Figure 4 . . . Local Network Physical Connections



Solid lines are user task inputs.

Dotted lines are user task outputs.

Figure 5 . . . Logical Flow In Local Area Network

## LOCAL COMMUNICATIONS (LC) MODULE

In the description which follows, "Sender and "Receiver" are generic names of a module which sends a message and the module which receives the message. The LC module will have the following properties:

- Acknowledgement of individual messages, i.e., only one message at a time will be outstanding between a pair of functional modules.
- Standard message format will have a place for acknowledgement. Whenever possible acknowledgements will be piggybacked onto a message travelling in the reverse direction. Upon receipt of positive acknowledgement, the sender will release the acknowledged message's buffer space. Upon expiration of a time out, the sender will re-transmit message. Retransmission will be repeated once, if the first retransmission also fails. If the second retransmission fails, the sender will notify the Recovery Management (RM) Module of the existence of an error condition between the relevant pair of physical nodes and the relevant pair of modules. The sender will send no more data to any module and the receiver will not process data from any module until notified by the RM that the error has cleared.
- Messages will be transmitted in one continuous stream of bits, i.e., messages will not be split into packets. This will simplify the communications protocol. This will require buffer space to be reserved for the maximum size message. However, buffer size requirements will tend to be reduced because no more than one message will be unacknowledged at a time. No message greater than the maximum size message will be transmitted by a module. In the rare case where a message is larger than the maximum size message, it will be broken into two or more smaller fragments.

It will be possible to set up a virtual circuit between any two modules (Figure 6). We don't say between any two nodes because a node could contain two or more modules. The virtual circuit would be implemented by creating tables in the Session Services Module at the sending and receiving ends of the connection.

The following types of communication will be possible (Figure 6):

## LAN

### Foreground

Virtual circuit between two functional modules residing in SPLICE minicomputers (e.g., virtual circuit between Terminal Management and Data Base Management Modules)

### Background

Virtual circuit between a functional module residing in a SPLICE minicomputer and a functional module residing in a main frame (e.g., Terminal Management Module and Burroughs DBMS)

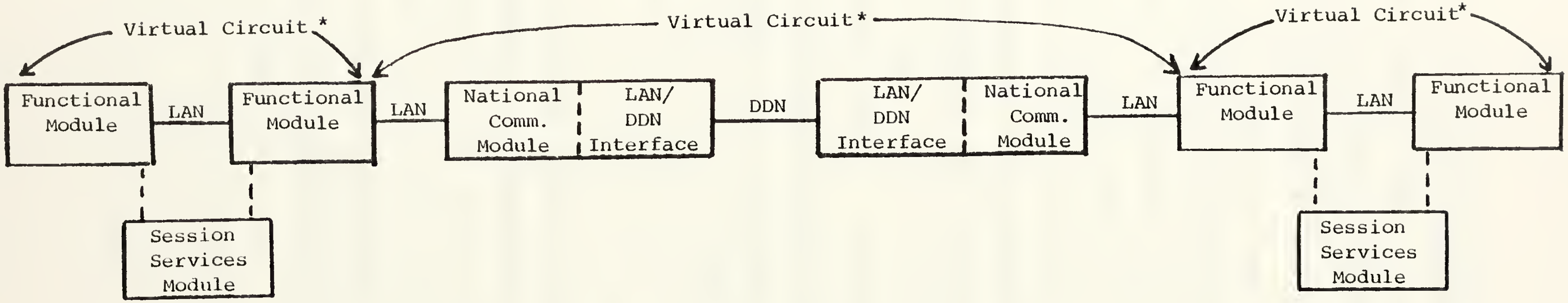
## DDN

### Foreground

Virtual circuit between a local functional module residing in a SPLICE minicomputer and a remote functional module residing in a SPLICE minicomputer.

### Background

Virtual circuit between a local functional module residing in a SPLICE minicomputer and a remote functional module residing in a main frame.



* Could be foreground or background connection.

Notes: 1. For ease of explanation, only two functional modules are shown in each LAN.

2. Solid lines indicate message path; dotted lines indicate support function.

Figure 6 . . . Virtual Circuit Possibilities

## Addressing

In order to implement the architecture which has been suggested, it will be necessary to assign logical addresses to the various functional modules which will be contained within the SPLICE minicomputers and to the functional modules which currently exist in the SP and ICP main frames (e.g., Burroughs computers). The latter requirement will necessitate identifying which programs or packages in the main frames (e.g., data base management) consitute functional modules. Furthermore, it will be necessary to create and maintain a table which will return the physical address of a hardware unit when the logical address is provided as an argument [4]. This table and the associated maintenance functions will be the responsibility of Session Services. The use of logical addresses will allow mobility of functional modules, i.e., it will be possible to relocate a functional module to a different hardware unit when required for performance or recovery purposes.

## Network Layers

For sending and receiving messages on the DDN, all layers of the ISO model will be used, as shown in Table 2. The Data Link and Physical layers are shown unspecified in Table 2. The LAN has no need for the services normally provided by the Transport and Network layers. Additionally, the application layer is shown for the purpose of completeness; it has no effect on message handling within the LAN. The Presentation layer, implemented in the Terminal Management Module, will accept data from the application process and convert it to the standard LAN format. Conversely, it will accept messages in standard LAN format and convert them to the appropriate application process format. A terminal user is considered to be one of the "application processes".

For the purpose of simplifying the LAN design and in view of the fact that intra LAN communication does not require all seven ISO layers, but DDN communication does require all layers, the following message formats will be used:

- TCP format [5] will be provided to the DDN by the NC module (described in the next section) whenever communication on the DDN is necessary. A much simpler format, as shown in Figure 7, will be used for intra LAN communication [14].
- End to end virtual circuit connections and breaking of complete messages into fragments, services normally provided by the transport layer, will be implemented in each of the functional modules. "End to end" in this context refers to the logical communication linkage between two modules which are separated by a relatively short distance; in some cases, the two modules could be in the same hardware unit. The above approach is considered to be the best solution to the dilemma of whether to include TCP in the LAN communication. If it were included, functions which are not needed, (e.g., flow control) would be implemented unnecessarily. On the other hand, if we were to modify the SS layer to incorporate these needed TCP functions, the LAN SS Layer would differ from the SS Layer which is needed for the DDN (where the complete TCP is utilized). It is appropriate to use a subset of the DDN virtual circuit protocol which is as close to the long haul network protocol as possible, rather than design two separate protocols [13]. This approach makes translation between the two protocols easy and provides for protocol compatibility.

Table 2

Use of ISO Layers in LAN Design

LAN Communication

| Layer | Module |
|---|---|
| Application | Application process modules (e.g., APADE, IDA)* |
| Presentation | Terminal Management |
| Session | Session Services |
| Data Link | Local Communications |
| Physical | Local Communications |

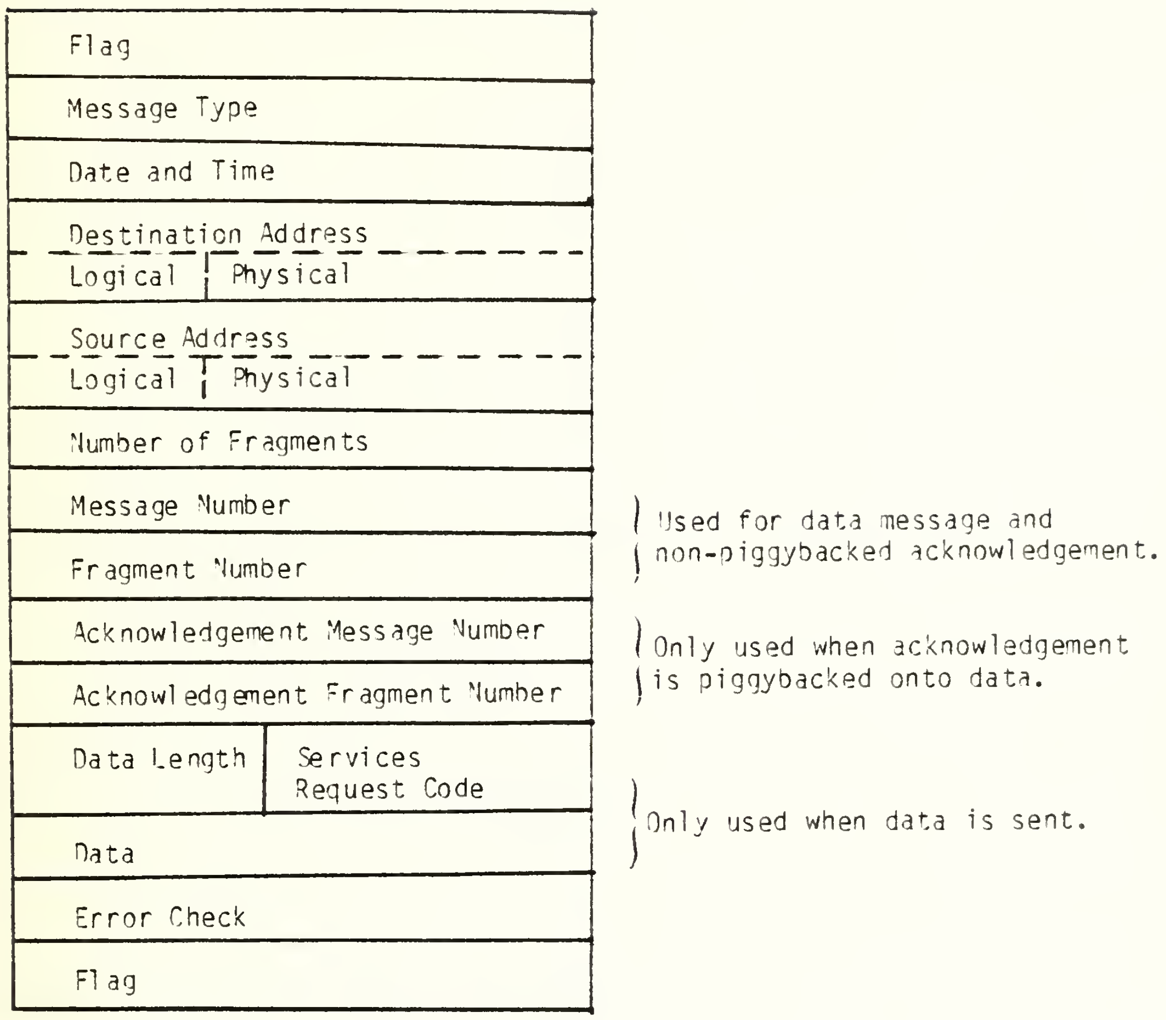
DDN Communication

| | |
|---|---|
| Application | Same as for LAN |
| Presentation | Terminal Management |
| Session | Session Services |
| Transport | TCP |
| Network | IP |
| Data Link | Specified by the DDN |
| Physical | |

* Application processing and data base management should be incorporated into the functional modules (e.g., Terminal Management and Data Base Management) as much as possible.

## Message Formats

Intra LAN message formats are illustrated and defined in this section (Figure 7). LAN message formats have been suggested by many authors including [5]. Acknowledgements will be piggybacked onto data messages whenever data is ready to send to the module which requires an acknowledgement (Figure 9). In those cases where there is no data to transmit, an "ordinary acknowledgement" will be used (Figure 8). This procedure requires provisions for identifying both new messages and acknowledged messages. In order to account for the possibility of a long message which exceeds the maximum buffer size of a functional module, the concept of "fragment" is used. A fragment is simply part of a message. This technique requires identification numbers for both messages and fragments, for new messages and acknowledged messages (Figure 7).


Flag
Message Type
Date and Time
Destination Address
Logical
Physical
Source Address
Logical
Physical
Number of Fragments
Message Number
Fragment Number
Acknowledgement Message Number
Acknowledgement Fragment Number
Data Length
Services Request Code
Data
Error Check
Flag
Used for data message and non-piggybacked acknowledgement.
Only used when acknowledgement is piggybacked onto data.
Only used when data is sent.

Note: All fields are fixed length except data portion.

Figure 7 . . . LAN Message Format

Flag: Bit pattern which signifies the beginning and ending of a message fragment.*

Message Type: Code indicating type of message:

- Normal Data (FIFO)
- Priority Data
- Ordinary Acknowledgement (Figure 8)
- Data with Piggybacked Acknowledgement (Figure 9)
- Reset LAN (Reset LAN communications after error condition. Purge messages in transit. Reset message counters to zero.)
- Reset message and Fragment Counts (Reset counters to zero.)
- LAN shutdown
- etc.

Date and Time: Day, month, year and 24 hour clock time of transmission from sender.

Destination Address: Logical and physical addresses of sending module.

Source Address: Logical and physical addresses of sending module.

Number of Fragments: Number of fragments contained in a message. Used for message sequencing and acknowledgement control and by the receiver for allocating buffer space.

Message Number: Sequential number assigned to each transmitted message. In the case of an acknowledgement, the number of the message being acknowledged is placed in this field. The number is reset to zero periodically. Each module will be responsible for setting, incrementing and resetting this

* A fragment is either an entire message or part of a message which is too long to transmit as a single entity.

count. Thus the receiver will know which message number it should receive next and the sender will know which message number should be acknowledged next by the receiver.

Fragment Number: Sequential number which is assigned to each fragment comprising a message. This number is reset to zero by the sender when the first fragment of the next message is to be transmitted. Each module will be responsible for setting, incrementing and resetting this count. Thus the receiver will know which fragment number it should receive next (The message number should increase after all fragments have been received). The sender will know which fragment number should be acknowledged next by the receiver. The first fragment will be numbered Ø.

Acknowledge Message Number: Message number that is being acknowledged.

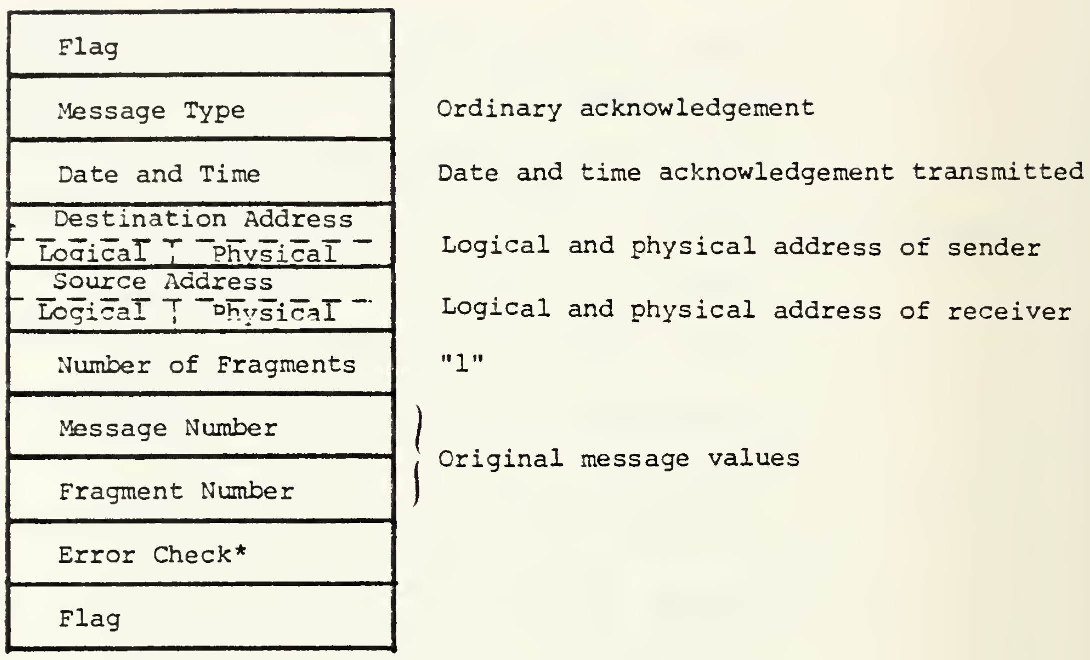
Acknowledge Fragment Number Fragment number that is being acknowledged.

Data Lenth: Number of bytes contained in the data portion of a fragment. This value can be zero.

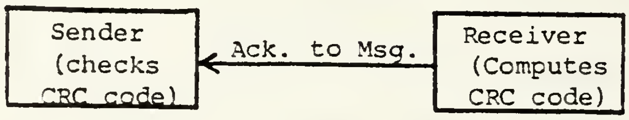
Services Request Code: Code which indicates which service (e.g., retrieve record) is desired by a process (e.g., user task).

Data: User or control data.

Error Check: CRC code computed by sender. If an error is detected by receiver, the fragment is discarded by receiver and no acknowledgement is sent to the sender. After appropriate time-out, sender will retransmit. If this attempt fails, the Recovery Management (RM) module is notified of the existence of an error condition.



Note: In an ordinary acknowledgement, the roles of "sender" and "receiver" are reversed, as shown below.

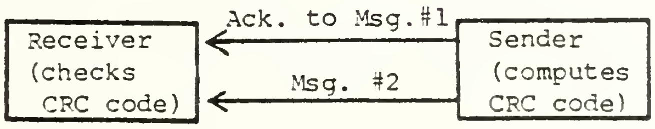


* CRC code computed by receiver. If error detected, sender resends fragment. Receiver has duplicate fragment detection capability. If duplicate detected, it is discarded, acknowledged and original fragment is processed.

Figure 8 . . . Ordinary LAN Message Acknowledgement

| | |
|---|---|
| Flag | |
| Message Type | Acknowledgement piggybacked onto data |
| Date and Time | Date and time data and acknowledgement transmitted |
| Destination Address: Logical \| Physical | Logical and physical address of sender |
| Source Address: Logical \| Physical | Logical and physical address of receiver |
| Number of Fragments | |
| Message Number | Message number of data |
| Fragment Number | Fragment number of data |
| Acknowledgement Message Number | Message number of data being acknowledged |
| Acknowledgement Fragment Number | Fragment number of data being acknowledged |
| Data Length \| Services Request Code | |
| Data | |
| Error Check* | |
| Flag | |

Note: In a piggybacked acknowledgement, the roles of sender and receiver are as shown below.



* CRC code computed by sender. If error detected by receiver, it will discard fragment and retransmit message #1 after appropriate time out. Sender will detect duplicate, discard it, transmit acknowledgement and use original fragment. Sender will retransmit message #2 after appropriate time out.

Figure 9 . . . Piggybacked LAN message acknowledgement

## Control and Data Busses

As shown in Figure 3, there are logical control and data busses. It was mentioned that this virtual architecture probably will not be implemented physically. However, it will be implemented logically by providing separate ports or addresses in a functional module for addressing control and data messages. In addition, each message type will have its own queue and the control message queue will have higher priority than the data message queue, i.e., if control messages exist in its queue, all of them will be processed before any data messages are processed. The reason for this is that there could be control messages pertaining to system shutdown, recovery error conditions, etc. which require the immediate attention of the module. The implementation of a virtual control bus is in accordance with the principle of separation of functions in software design, i.e., grouping like functions in the same module. This procedure would allow changes to be made in control message functions and formats without affecting data message software, and vice versa. Also, the amount of control message traffic could be significant. System loading can be alleviated by allocating separate queues to the two message types and by physically implementing data message and control message ports at the node level with separate DMA channels.

## NATIONAL COMMUNICATIONS (NC) MODULE

Very roughly, computer networks can be divided into two distinct types: local area network (e.g., SPLICE LAN) and long-haul (e.g., DDN). Increasingly, networks are no longer limited to either type. An emerging requirement is to interconnect the two types of networks. As described in [7], the interconnection problem may be viewed in terms of interfaces and network services. Each of these may be divided according to whether they possess datagram or virtual circuit characteristics. A datagram interface allows a user to enter packets into the network independent of other packets; each packet will be handled separately. A virtual circuit interface requires that an end to end logical circuit be established between source and destination. A datagram service is one in which each packet is "on its own"; sequenced delivery or any kind of delivery, for that matter, is not guaranteed. Duplicates are possible. Virtual circuit service, on the other hand, guarantees sequenced delivery, provides flow control and eliminates duplicate packets.

In the NC module of the LAN, both sides of the interface will provide a virtual circuit service. Message fragments are transmitted within a LAN and packets are transmitted on the DDN backbone. The NC module is responsible for providing the interface between each LAN and the DDN. In particular, this module will provide the conversion between the LAN protocol and TCP and vice versa. Instead of connecting all LAN modules and nodes directly to the DDN, a gateway is provided [7,13]. The NC module, located in the Front-End Processor (FEP), provides protocol conversion and the Interface Message Processor (IMP) of the DDN, to which the FEP is connected, serves as a gateway (see Figure 10).

Packets which constitute a message from the DDN are accumulated at the

NC in a manner similar to that which occurs at the destination IMP in ARPANET. A message or fragment (one of the functions of NC is to perform message fragmentation on incoming messages, where neccessary) is then sent by the NC to the destination module on the LAN. The SS at the originating LAN would have recorded the destination module logical address and physical address in the message (see Session Services in next section). It will be a function of the Network Management (NM) Module located at FMSO to broadcast changes in physical addresses to the various LANs. The NM will maintain uniqueness of logical and physical addresses by assigning them to the various LANs (Recovery Management Modules). (The detailed functions of the NM are not covered in this report). It will be the responsibility of the Recovery Management (RM) Module in each LAN to maintain the LAN copy of the Network Directory, i.e., make changes to network physical addresses. The SS Module will have read but not write access to the directory.

Physically, NC will reside in the FEP. Unless a message requires priority handling, it will send messages to the nearest Packet Switching Node (PSN), or IMP, of the DDN on a FIFO basis. Since the speed of the LAN is greater than the DDN, the NC buffer space could become exhausted [7, 13]. This will be handled, as in the LC, by only allowing a single message from a given Functional Module (FM), to be unacknowledged at a time and by reserving a buffer space equal to the maximum size message fragment. This approach will also have the advantage of providing a uniform method of message handling, independent of whether the message is intra or inter LAN.

Only those aspects of the TCP which are necessary to convert messages from LAN to DDN format and vice versa will be implemented in the NC.

## Brief Description of TCP [5]

The key characteristics of TCP are the following:

° Host-Host Protocol (i.e., end to end protocol)

° Located in ISO Transport Layer

° Guaranteed sequenced message delivery

° Logical full duplex connections

° Sequence number assigned to each octet (8 bits)

° Time out used to control message sequencing and acknowledgements

° Connection name used to refer to connections after the connection has been established. This would be the concatenation of the SPLICE logical address (functional module address) and the physical unit address.

° Users may indicate precedence and security of messages.

° Window oriented flow control

° Message segments reordered at destination TCP

° Acknowledgements required

## TCP Functions Implemented by the NC Module

The following TCP User Commands will be implemented in the NC, where "User" really means "NC". The terminal user will employ a command language which TM will interpret and send along to NC when a remote module is to be utilized. See Reference [5] for a complete description of these commands.

° OPEN

- Active: Begin procedure to synchronize the connection at once.
- Passive: LISTEN for an incoming connection.

The local and remote NCs will notify their respective pertinent modules when a connection has been established.

° SEND

  - Send data contained in the indicated user buffer on the indicated connection.

° RECEIVE

  - Allocate a receiving buffer associated with the specified connection.

° CLOSE

  - Close the specified connection

  - The local and remote NCs will notify their respective pertinent modules that the connection has been closed.

° STATUS

  - Obtain status of connection (e.g., OPEN or CLOSED). According to [5] it is not mandatory to implement this command. However, this would be a very worthwhile command to implement for SPLICE.

° ABORT

  - Causes all pending SENDs and RECEIVEs to be aborted. A RESET message is sent to the remote TCP. The local and remote NCs will notify their respective pertinent modules when a connection has been aborted.

The NC will be able to request network status (e.g., PSNs up/down, SPLICE processors up/down, etc.) from the NM which, it is assumed, will be able to obtain DDN status information from the network management facility in the DDN. As far as LAN status information is concerned, it will be the responsibility of the RM to forward changes in LAN status (e.g., SPLICE processor down) to the NM and to all functional modules and processors within its LAN.

## Interpretation of TCP Address

The TCP uses a port identifier in its header [5]. The concatenation of the port identifier with the network and local addresses used in the Internet Protocol Message Header constitutes a socket. The Internet Protocol is the DOD standard network layer protocol [8]. The local socket refers to the identification of the source (process initiating connection) end of a connection and the foreign socket refers to the identification of the destination end of a connection. After a connection has been opened, the local-foreign socket pair may be referred to by a connection name. In the LAN, port identification corresponds to the logical address of a functional module. The network address and local address corresponds to the LAN and physical processor in which the module resides, respectively (see Figure 12). As stated previously, logical addresses do not change, whereas physical addresses are subject to change, allowing for mobility of modules. Both types of addresses are necessary in order to access a particular functional module in the SPLICE network.

## Interconnect Between LANs and DDN

As indicated previously there are two basic ways of providing communication between two processes: virtual circuits and datagrams. The former provides reliable end-to-end communication between a pair of processes, with sequenced message delivery and elimination of duplicate messages ensured. In order to establish a virtual circuit, as with the X.25 Protocol, a set up procedure is necessary [18]. This is the call request, which, in the X.25 Protocol, establishes a fixed route between gateways (interfaces between two networks) for the virtal circuit; routing within the networks which are connected by the gateways may be dynamic (i.e., non-fixed) [11]. Virtual circuits may also be divided into the two categories of switched and permanent [18]. The former is implemented when the called process accepts the call request issued by the calling process. A permanent virtual circuit, on the other hand, is always established; hence, it does not require a call-up procedure.

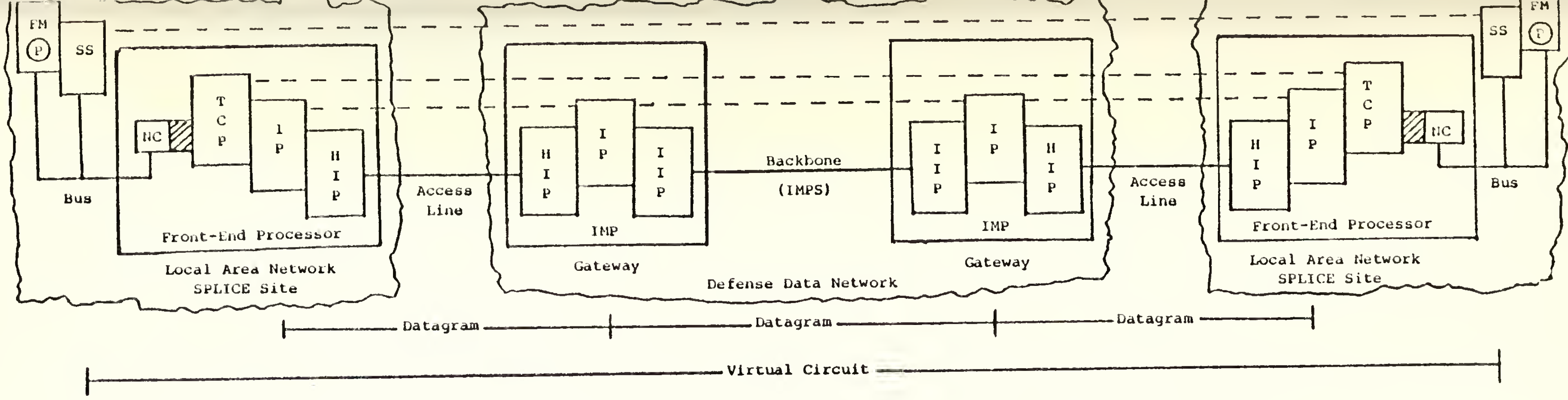
The datagram form of transmission provides a transaction service (e.g., update a file record) [11]. The Transmission Control Protocol [5] of the DDN which corresponds to the Transport Layer of the ISO OSI Model [24], provides virtual circuit service [11,19]. The Network Layer of the ISO model, as implemented in DDN, will be the Internet Protocol [8]. The Internet Protocol will provide a datagram service operating under TCP. This concept is shown in Figure 10, where each LAN would be connected to the first Packet Switching Node (PSN) of the DDN. As shown in Figure 10, virtual circuits are implemented by placing the modules (e.g., Session Services) and protocols (e.g., TCP) at the end points of the network [21]. In order to transport a message of the format shown in Figure 11 across the DDN, it must first be encapsulated in the TCP header which, in turn, will be wrapped in the Internet header as suggested by Figure 11 [21]. This encapsulated data is called a datagram. Datagrams exist between points in the network where the Internet Protocol (IP) is located, as shown in Figure 10 [20]. The IP will examine the network address portion of the datagram at the gateways (see Figure 10) and route it to the destination LAN gateway. Thus the IP and its datagram form of communication provide transmission in the communications subnet (i.e., DDN), whereas TCP and its segment provide the end-to-end virtual circuit service.

Some networks - those employing X.25 and SNA - employ fixed routing [23]. This procedure guarantees sequenced message delivery. The DDN, by using two protocols - TCP in the transport layer and IP in the network layer, can provide virtual circuit service (TCP) without requiring fixed routing. The Internet datagram service provides dynamic routing so that all datagrams corresponding to a given session do not have to follow the same fixed route.

In many networks a distinction is made between application modules and their names and frequently accessed server modules (e.g., file server) and their names [22]. In the proposed design the "application modules" are the server modules (e.g., functional modules). The functional module names are indicated as the logical addresses in the message part of the datagram in Figure 12. These addresses are copied into the TCP header. The physical node addresses in the message section (See Figure 12) are copied into the Internet header. Because there could be more than one FEP and set of physical nodes associated with a given LAN, a LAN address and physical node address (the address of the node containing a process) are required in the Internet header.

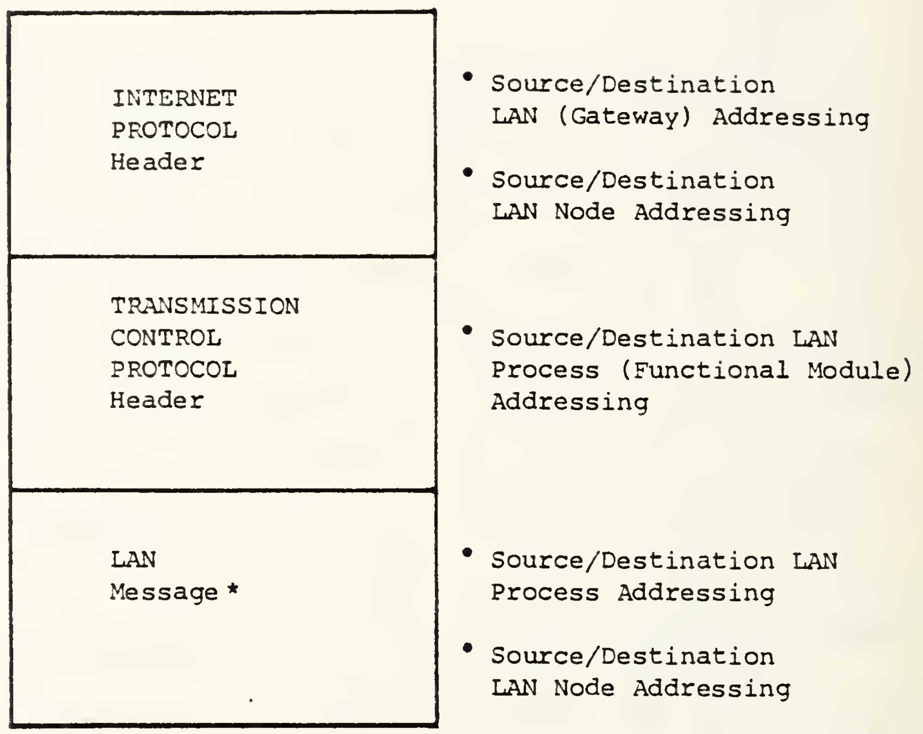
It is the responsibility of the National Communications (NC) Module in the Front-End Processor (FEP) to get an incoming message routed to the correct functional module (FM) on the LAN when a datagram is received over the DDN and to prepare an outgoing message in the format of Figures 11 and 12 for transmission on the DDN. It will also be the responsibility of NC to fragment both incoming and outgoing messages, if necessary, in order to be compatible with the maximum size buffer of the FMs and to reassemble incoming message fragments. Each outgoing fragment will be put into the format of Figure 11.

All acknowledgements over the DDN are done between the FEPs. Once a message is delivered to a FEP from the DDN, a one-for-one (i.e., stop and wait) acknowledgement system will be used between the addressed FM in the LAN and the National Communication Module in the FEP Similarly the stop and wait acknowledgement method will be used between the source FM and its associated FEP on an outgoing message. For both incoming and outgoing messages, fragments could exist and, when this is the case, acknowledgements within an LAN will be done on a fragment basis. Once an outgoing message fragment is acknowledged by NC to the FM, it is released to be sent over the DDN as a datagram, with acknowledgement of the corresponding TCP segment occurring between the source and destination TCPs. Similarly, once an incoming TCP segment, wrapped as an Internet datagram, has been acknowledged between the pair of TCPs, it will be sent to the intended FM and acknowledged as a fragment to NC.



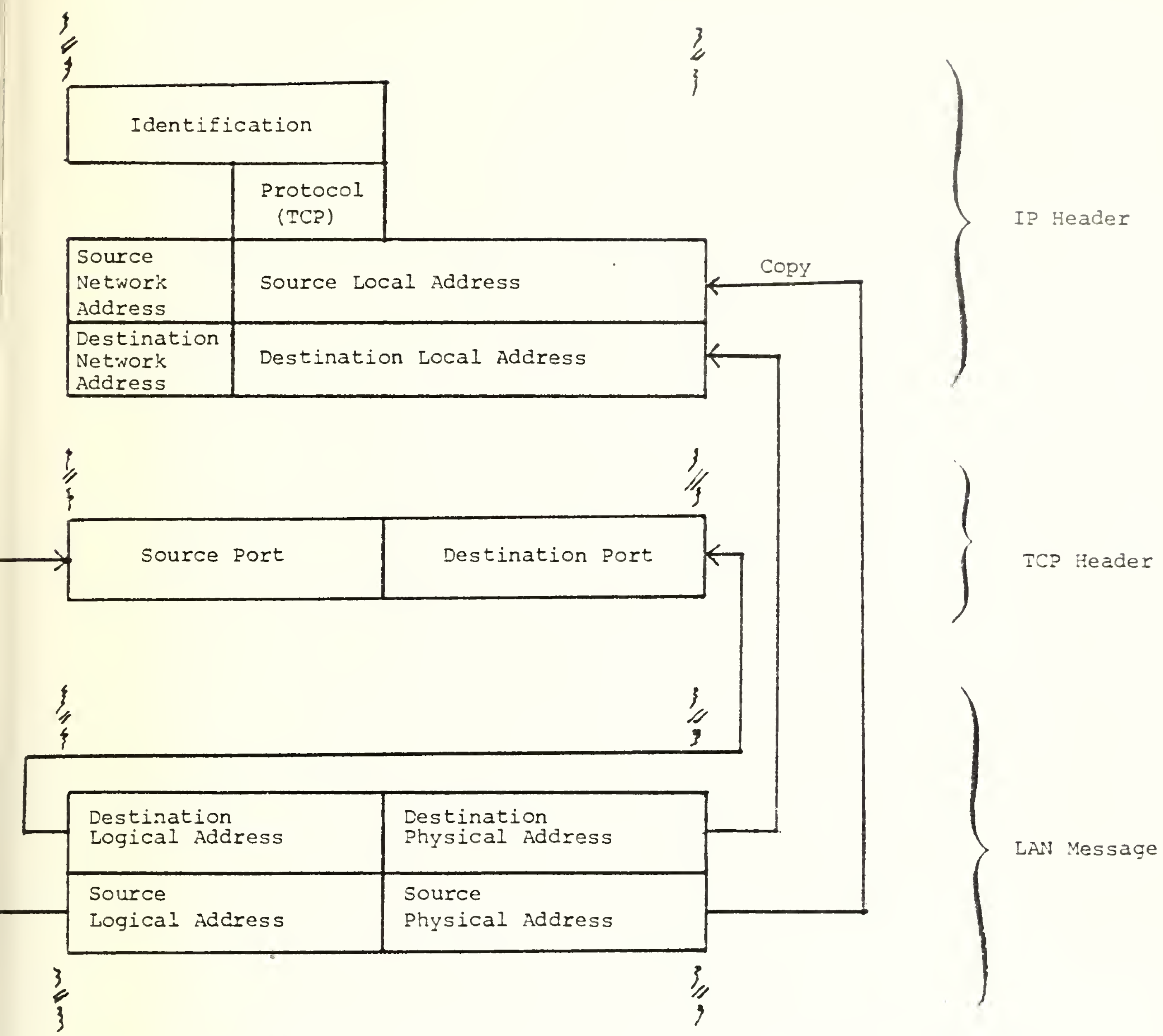
| | |
|---|---|
| P: | Process |
| FM: | Functional Module |
| NC: | National Communication Module |
| SS: | Session Services Module |
| TCP: | Transmission Control Protocol |
| IP: | Internel Protocol |
| HIP: | Host-IMP Protocol |
| IIP: | IMP-IMP Protocol |
| ▨ : | Protocol Conversion |

Figure 10. . . Relationship Between Local Area Networks and Defense Data Network


INTERNET PROTOCOL Header
TRANSMISSION CONTROL PROTOCOL Header
LAN Message *
• Source/Destination LAN (Gateway) Addressing
• Source/Destination LAN Node Addressing
• Source/Destination LAN Process (Functional Module) Addressing
• Source/Destination LAN Process Addressing
• Source/Destination LAN Node Addressing

* Complete message or fragment

Figure 11 . . . Internet Datagram



Identification: Identifier used By IP to Reassemble All Fragments Belonging to a Datagram (Value Provided by TCP)

Protocol: Identifies Next Protocol Above IP (i.e., TCP)

Source Network Address: Address of Source LAN

Source Local Address: Address of Source LAN Physical Node

Destination Network Address: Address of Destination LAN

Destination Local Address: Address of Destination LAN Physical Node

Source Port: Source Logical Address

Destination Port: Destination Logical Address

Figure 12 . . . Internet Datagram Showing Fields of IP Header, TCP Header and LAN Message which are Relevant to Addressing

## SESSIONS SERVICES (SS) MODULE

Figure 13 shows how the Session Services (SS) module will be used to coordinate the interaction among a user task, TM and another functional module (e.g., DBMS). A services request code in a message, corresponding to a user request task, is used by SS to obtain the logical and physical addresses of the functional module which will perform the requested service. Session Services invokes the appropriate functional module which, in turn, accesses the user message in the TM buffer. The functional module returns the required data after interpreting the instructions in the user message. It is a function of TM to present the data provided by the functional module in the format desired by the terminal user. The number in the figure corresponds to the sequence of steps in the process. It is evident that the following ISO layers are involved:

Application: User request task

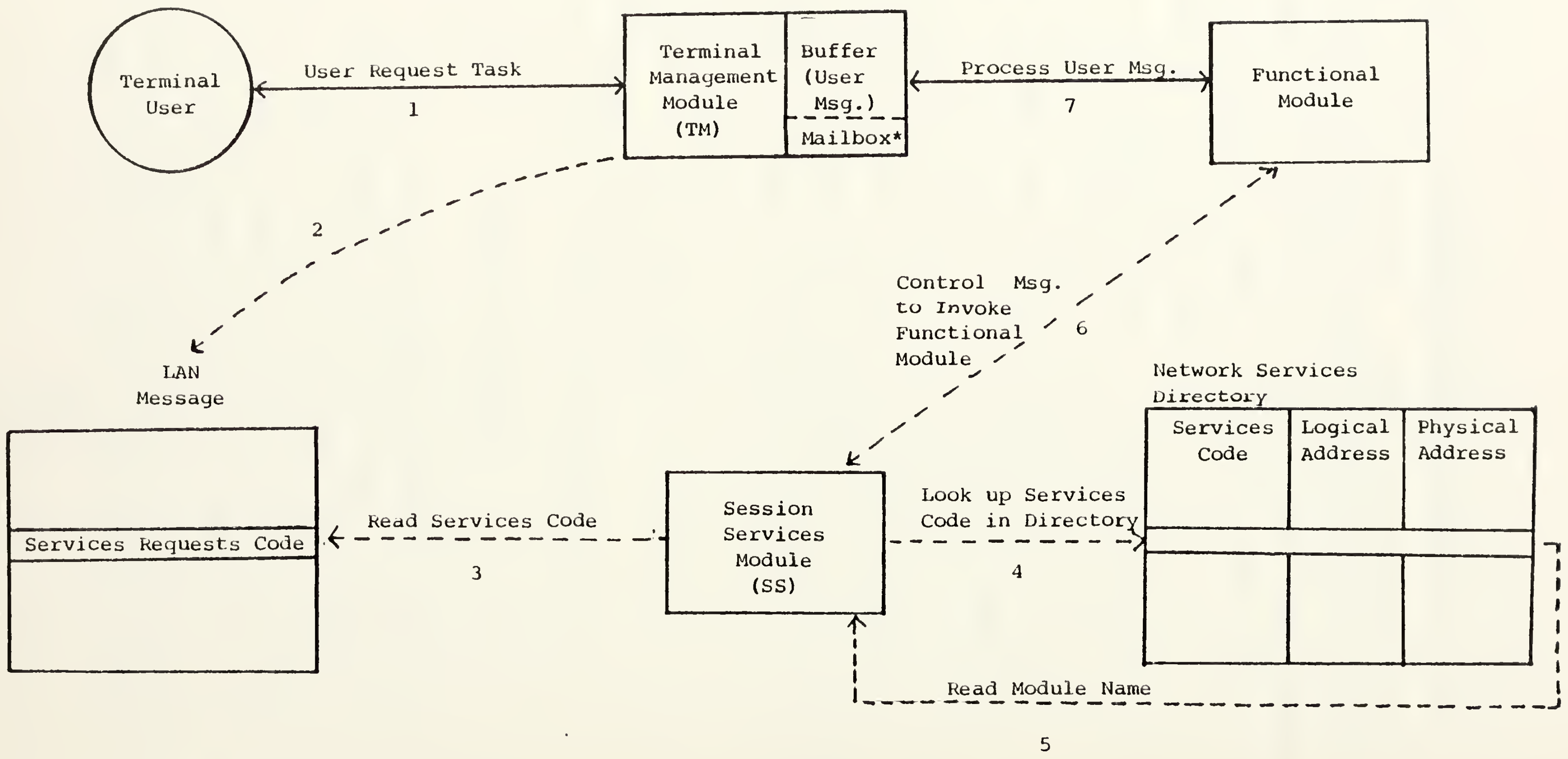
Presentation: Formatting by TM

Session Services: Invoking the functional module

$\left.\begin{array}{l}\text{Data Link}\\ \text{Physical}\end{array}\right\}$ Message transmission

It will be noted that Transport and Network layers are not used.

Terminal Management (TM) is the primary interface with the user process. Session Services coordinates and controls the various functional modules (whether on LAN, local hosts or remote hosts) to provide the services required by the user process [26]. The functional modules act as servers to session services for performing various tasks (e.g., data retrieval). Figure 13 shows Session Services operating through TM. This is because user process messages will be resident in TM during a session and because TM will keep the terminal user informed of the progress of processing his request. Session Services will issue various messages to the functional modules, such as "get record x", "print record x", etc.



° Dotted lines indicate control messages.

° Solid lines indicate data messages.

* Mailbox for non-interactive work.

Figure 13 . . . Session Services Coordinating Function

## Mail Box Capability

There is no inherent reason why a terminal user should have to be logged on or sit at a terminal through what could be long sessions while his data is being processed. There are many operations, usually those involving complex data management functions, which are not the short, interactive type. For those transactions which do not require continued user attention and interaction with the LAN, the transactions will be input by the user, and deposited in a mailbox [25]. Physical implementation of the mailbox may require disk storage in addition to main memory in the minicomputer where Terminal Management is resident. Session Services, upon receiving a control message from Terminal Management, as depicted in Figure 13, will be responsible for setting up a "session" between the mailbox and a functional module (FM). The FM will read messages from the mailbox and process them during the time the user is logged off. The FM will return any results to the mailbox for pick up by the user during a subsequent terminal session.

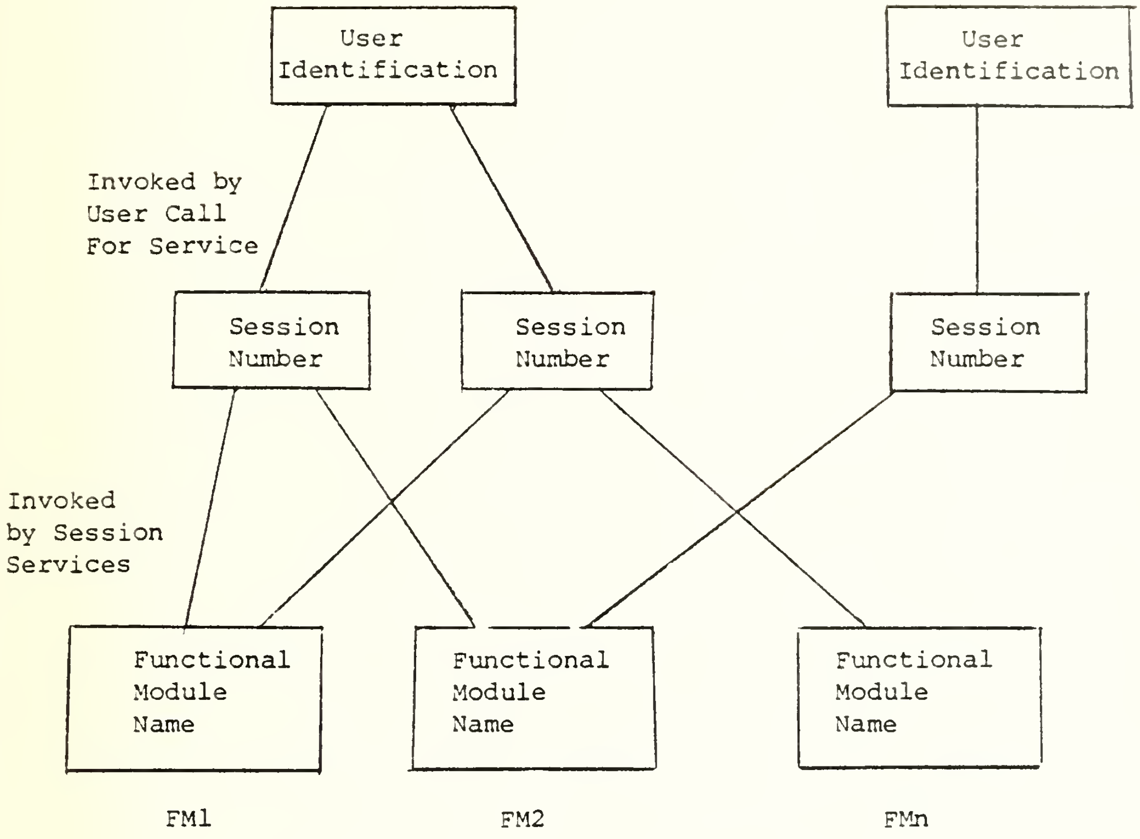
It should be noted that either a local or remote FM could be involved in either interactive or non-interactive processing and that in some situations it will be necessary for SS to invoke more than one FM to process a user task (e.g., the Data Base Module for retrieving data for terminal display and Peripheral Management for producing printed output). Also, in general, an FM could be local (e.g., local Data Base Module returns data to be displayed on local terminal) and another FM could be remote (e.g., locally retrieved data is sent over the DDN to be displayed via Terminal Management on a remote terminal).

## Multiple Sessions

The various sessions which can be conducted require a data structure for keeping track of these sessions and the functional modules (FM) which take part in a session. A tree data structure for this purpose is shown in Figure 14. Each time a user request is analyzed by SS and it determines which FMs are involved (one or more), a unique session number is assigned. The session number will be the mechanism whereby SS can determine which modules are involved in subsequent message transmissions for a given session and the sequence of message transmission and sequence of using the FMs (e.g., record extraction by Data Base Management and subsequent print of the record by Peripheral Management). The following should be noted in Figure 14. FM can be involved in more than one session at a time, involving more than one user:

° A user can have more than one session at a time.

° A session can involve multiple FMs.



Functional Module Name ≡ Logical Address: Assuming Module Names are uniquely assigned throughout SPLICE system; otherwise, Network Address and Physical Address must be concatenated with Logical Address.

Figure 14 . . : Session Services Data Structure

## TERMINAL MANAGEMENT (TM) MODULE

As indicated previously, the major design approach is to provide a set of functional modules (e.g., data base management) each of which will provide the same basic service (e.g., retrieve a record for all applications). Using this approach, the place in the system where applications are differentiated is the human-machine interface, i.e., the terminal screen formats. In addition to this consideration, the Naval Supply System has been plagued by inflexible terminal operations with respect to speed, code, format, editing capability, line discipline and command language. Furthermore, existing terminals are heavily dependent on inflexible vendor (Burroughs) terminal control units and most of the terminals cannot be programmed for changing I/O requirements. Some of these I/O inflexibilities can only be cured by obtaining modern terminals and control units, where either or both can be programmed. However, equipment alone will not provide a long term solution to achieving flexibility of terminal operations. The Naval Supply System must not again be in the position of being a "captive" of its terminal characteristics. In order to avoid this possibility, a fundamental change in design approach is necessary. Two measures are recommended in order to correct current deficiencies.

### 1. User Designed Screen Formats

One measure is designed to provide the user with flexibility of terminal screen format. Despite the fact that efficiency of design is achieved by generalizing the "application" modules (i.e., functional modules), the user naturally wants screen formats which are application oriented. Since one of the major lessons which has been learned in data processing is that it is futile to try to anticipate user requirements - these frequently evolve in unpredictable ways - a wise strategy is to provide a capability for the user to do a certain amount of the design himself, namely, terminal screen design. This is accomplished by the user designing a template of screen format, along with the prompts which are desired. At first blush this approach may seem to be imposing an undue burden on the user. Further reflection may convince the reader that it has the following advantages:

° It is thoroughly consistent with the idea of providing generalized functional modules, i.e., the screen design feature would be a part of the Terminal Management (TM) module (alternatively, it could be part of the Data Base Management module). As such, it would be available to <u>all</u> users and applications, as opposed to the traditional approach of application programmers designing specific screen formats and the supporting programming for each application. Our recommended approach shifts the burden of providing this software from FMSO to a vendor provided package with the required capabilities. As mentioned previously, this approach is cost-effective because vendor development costs are spread across many customers and, in addition, significant application design and programming costs, involving the use of critical personnel skills, could be avoided by FMSO.

° This approach eliminates the usual never-ending effort to upgrade user terminal display and report capabilities by providing for inevitable future change by acquiring, during the acquisition phase, a software package with built-in capacity for change, rather than FMSO having to respond in fire-fighting fashion to changing requirements during the operational phase. Since the user becomes responsible for fashioning his own display and report formats, he is more likely to be supportive of the system or at least is less likely to be critical than would otherwise be the case. Both terminal display (screen) and report formats could be designed by the user. Incidentally, there is nothing in this proposal to prohibit FMSO analysts and programmers from designing these formats, should this be desired by FMSO management. Obviously, some centralized control over screen and report formats which are common to the Stock Points and ICPs is necessary in order to prevent the chaos which would result with uncontrolled user design of formats. On the other hand, those formats which are unique to a Stock Point or ICP could be designed locally. In this discussion the term "user" is employed generically and is not meant to imply that individual stock clerks, as opposed to supervised user organizations, would necessarily design formats.

The screen and report formats would be designed, using the terminal itself as the medium for specifying the desired formats, and stored by the system for subsequent recall by the user when performing file update and query operations. A tremendous advantage of this approach is the ability to permanently store the templates and to add, delete and change fields by using the vendor's data definition language, and to add, delete and reformat screen and report images by using cursor control formatting procedures at the terminal itself.

Logically, the vendor provided screen and report format design functions are part of Terminal Management and Peripheral Management modules, respectively, as defined in this report. However, depending upon the design of the vendor's software, these functions could be included in a data base management package or the report format design function might appear as part of a report generator. Except to the extent it affects modularity of design, the packaging of functions is essentially immaterial to the subject of this report, which is the functional specification of the LAN.

## 2. Virtual Terminal Capability

It has been found in a number of networks [9,10], notably ARPANET and European network developments, that a virtual terminal concept and its accompanying Network Virtual Terminal (NVT) protocol is necessary in order to accommodate a great variety of existing terminals in a network and, more importantly, to accommodate future terminal requirements. Where there are m hosts and n terminals, the resulting m x n problem can be reduced to an m x 1 problem [10] by using a standard terminal protocol - the NVT protocol - in the network. Each source terminal's characteristics are mapped to the NVT, for transmission in the network, and then mapped from NVT to the destination terminal's characteristics for presentation at that terminal, where "terminal" can be a computer process in addition to a physical terminal. Characteristics such as character code, line length, page width, print density, etc. are commonly included in the NVT. Of course, characteristics which are physically unavailable (e.g., specifying a line length of 132 characters on an 80 character terminal), cannot be achieved. The user employs the NVT by issuing commands, in the language of the NVT to "negotiate" source and destination terminal characteristics with the remote process.

The Defense Data Network (DDN) will have a ARPANET TELNET NVT feature [12]. In addition to the protocol required for LAN screen management, the NVT protocol will be a necessary part of the TM module, as indicated in Figure 4, in order to communicate with remote processes, where other than default terminal characteristics are desired for a given operation.

As indicated in the previous section and in Figure 13, the TM module will store user messages in its mailbox for subsequent processing by other FMs and will deliver result messages, deposited in its mailbox by other FMs, to the user's screen during a subsequent user terminal session.

## DATA BASE MANAGEMENT (DBM) MODULE

This LAN design provides for distributed control, as described in the next section, but does not provide for the distribution of data bases within a LAN. When viewed over the entire SPLICE system, data bases are obviously geographically distributed. For the purposes of maintaining adequate control over cataloging files and maintaining the integrity of related files in the data base (synchronization of updating procedures), the data base functions are centralized within each LAN. Other than the fact that some files will remain on the Burroughs Hosts and some files will migrate to DBM, there is no reason to provide for the distribution of data bases within a LAN.

### File Server

Processors will be dedicated to data base management (see Figure 15). It is arguable whether they could be classified as data base machines, since they will not have all of the specialized hardware which is usually associated with this machine [16]. Also, there is a difference of opinion among specialists concerning whether data base machines have achieved technical maturity [17]. The need for specialization of data base functions in the form of special purpose hardware, which is central to the design philosophy of data base machines, appears to be unnecessary in the case of SPLICE. Rather, a "special purpose software module" - a file server - residing in large scale minicomputers, with adequate fixed disk storage for highly active files, augmented by larger movable disks for less active files, should be adequate for handling foreground queries and file maintenance requests [17]. In a very loose sense, the file server acts as a data base machine, in that all data base processing is off loaded from other modules and node(s), and concentrated at a single module and node, but without the special purpose hardware of a data base machine (e.g., data base command and control processor, keyword transformation unit, index translation unit, security processor, etc.) [16].

This report does not take a position on whether the data base model should be network, hiearchical or relational. We think the capabilities of a vendor provided DBMS, in terms of dictionary, integrity, recovery, query language, etc. features, are more important than the particular model employed, although compatibility with existing COBOL programs would seem to argue for the CODASYL (network) approach.

## Functions of DBM Module

1. Catalog

   Maintain the catalog of file names and status:

   - Name
   - OPEN or CLOSED
   - Size
   - Physical address of file
   - Physical address of index
   - Application used in
   - Date entered into system
   - Expiration date, if any
   - Location of backup copy
   - Format
   - Access restrictions

   Only files pertaining to so-called foreground applications, (i.e., non-Burroughs files) will be cataloged. If a file name cannot be found, the message will be forwarded to the Burroughs node (unspecified at this time) which contains the catalog for host files.

2. Operations

Under a menu selection scheme, perform the following functions:

° Retrieve a record for terminal display.

° Change record in specified fields.

° Delete a record

° Insert a record

° Print a file

° Print a record

In the case of printing a record, the transaction message will be routed by TM to DBM first. The DBM module will locate and retrieve the record and send it to the Peripheral Management (PM) module for printing. With regard to printing a file, TM will also send the transaction to DBM first, which will open the file for PM. The PM module will access and spool the file for printing of these onto its own disk file. Printing can then occur without interference on the LAN, since PM disks will be local to that module's processor.

3. Dictionary

A dictionary will be provided for the purpose of defining and characterizing the data elements. The dictionary should be an integral part of vendor supplied DBMs. The dictionary will serve as an excellent vehicle for requiring users and developers to define data elements in SPLICE to bring definitions up to date, discard outdated elements, introduce new elements, and, in general, provide a method for rigorously defining data requirements. The dictionary will also be of great assistance for designing screen and report formats, as described under Terminal Management.

## DBMS Implementation

Terminal Management (TM) pre-processing of the user's query or update transaction and the post-processing of the returned data will be performed in one minicomputer (the one where TM is resident) while data base functions (e.g., record retrieval, record update, etc.) will be handled in a separate minicomputer. A proposed back end data base management system [15,27] utilizing the aforementioned specialized file server approach is shown in Figure 15. For high performance purposes, dbms overhead functions (e.g., dictionary, catalog and index) are handled by a separate processor. Among its functions is to index from the logical key field provided by the user process to a physical disk address. This processor has fixed head disk files for high speed operation of the above functions. The actual retrieval and storage of records and physical storage management is performed in a second data base processor, which has associated with it the actual data files. Active files are stored on fixed head disks; less active files are stored on movable head disks.

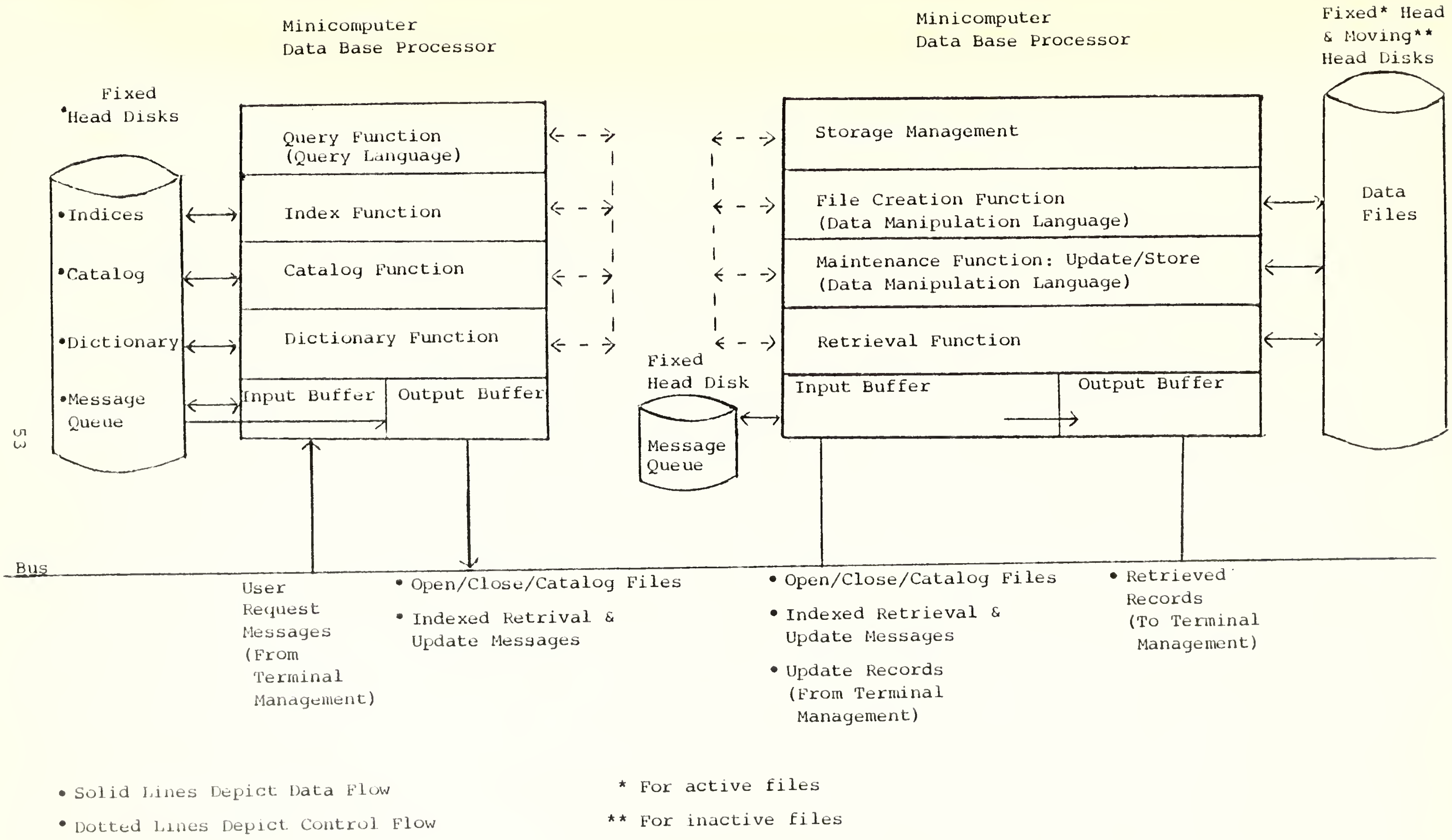


Figure 15 . . . Back-End Data Base Management System

## SYSTEM CONTROL

With the exception of Recovery Management (RM) and certain resource allocation functions, the various functional modules (FMs) operate as a distributed system. What this means specifically is the following:

- Modules exchange messages directly without first obtaining permission of, or passing them through a central control.
- FMs bid directly for the use of additional reusable resources (e.g., memory, fixed head disks) which may be necessary for providing greater workspace in order to conduct multi-tasking. Existing system and user files and the opening of new permanent files are not included in this category of resourses, being the responsibility of the Data Base Management module.
- A small Resources Allocation (RA) module assists the FMs in obtaining sharable reusable resources. This module is associated with the Resources Status Table (RST) of Figure 3. The RA, which is implemented in a dedicated processor, performs the function of shared resource allocation. It has available to it memory and fixed disk units which it allocates to the FMs on a shared basis. Naturally, FM processing which involves the use of shared resources will be slower per transaction than processing which uses dedicated resources because of transmission delays on the LAN and contention for shared resources. However, throughput would be increased.

This philosophy of control is amplified further below:

- Each FM will respond positively to only a specified set of Services Request Codes (e.g., retrieve a record for the Data Base Management module), as indicated in Figure 7. If the module receives an invalid

Service Request code, it will forward the message to the Recovery Management module for disposition, with an error code indicated.

Each FM will be equipped with sufficient dedicated resources (e.g., local memory) to be able to process its worst case Service Request. This means that a module will always be able to process at least one transaction, i.e., service request. This also implies that deadlocks cannot occur.

° A module only bids for the use of shared resources (See Figure 3), when it is presented with multiple tasks to perform and is unable to process them concurrently without the use of additional (shared) resources. The FM sends a resource request to RA, via a control message on the virtual "control bus", giving it the type and quantity of resource desired. In some cases multiple resources will be required. The RA module sends "grant" message to the FM if the resource is available in the quantity desired; RA will then subtract the acquired amount of resource from the available quantity of resource in the RST. Upon receiving a "grant" message, FM will set an interval timer to a system-specified maximum value. Upon the expiration of this interval, the FM returns the resources to the pool by sending a "release" message to RA. The RA module then adds the released resource to the quantity available in the RST. The timer interval length can vary among FMs, depending upon processing priorities, and can be set by the System Operator. An FM must bid again, if resources are required subsequent to the release of resources.

If the resource is not available in the desired quantity, the RA sends a "denied" message to the FM, which will continue to process with the use of local resources only. No record is kept by RA of this bid, and the FM must rebid at a later time, which is determined by an interrupt generated by a system-specified and operator adjustable timer interval set by the FM.

- All of the above descriptions pertain to the use of control messages flowing on the "virtual" control bus. Once an FM has acquired a resource, it will send data (file records in some cases) to be stored in the resource unit and read data from the resource unit, with the assistance of RA. Data messages will be transferred on the "virtual data bus" and will be addressed to RA according to the two level address procedures described previously, i.e., by the node in which RA resides and by the name of the RA module. In addition, RA must map between the message identification, as stored in a message by the FM, and the physical shared storage space. Upon receipt of a control message from the FM, giving the message identification, RA will map to the physical storage locations, retrieve the message and send it to the FM.

Although the above "contention system" of allocating shared resources is crude in that resource requests are not queued, and requests will not necessarily be served on a FIFO basis, it has the great advantage of simplicity and low cost, due to the self-regulatory nature of the scheme. As soon as recordkeeping and queue maintenance are introduced to keep track of multiple requests - order of receipt, type and amount of request - the complexity rises rapidly. It is possible that sufficient local resources could be economically provided to each FM, such that an overload would rarely occur, and shared resources could be dispensed with entirely.

## FUTURE WORK

The next step in the LAN design effort is to use the functional design specifications, which have been provided in this report, as the basis for designing a simulation model, which will be used to estimate the performance of the LAN in terms of response time and message transit time operating under a variety of transaction loading distributions. Quantitative data which are available pertaining to types, numbers and rates of transaction inputs will be used to drive the simulation. Quantitative analysis is vital to the design effort since, at this point, we have a qualitative design which we feel is workable, but significant quantitative performance analysis is required in order to provide greater assurance of the validity of the concepts.

Secondly, a mapping will be performed between the functional LAN design described in this report and a specific LAN physical system (i.e., topology, access method, communications medium, protocols and operating system). This will be done in order to select that hardware and software system, from among the available LAN alternatives, which best realizes the functional design.

## REFERENCES

1. K. Yada and T. Ochiai, "Optical Fiber Makes Research Information Processing System", Digest of Papers, Spring, Compcon 81, IEEE Computer Society Press, February 1981, pp. 450-453.

2. Edward P. Stritter, Harry J. Saal and Leonard J. Shustek, "Local Networks of Personal Computers", Digest of Papers, Spring Compcon 81, IEEE Computer Society Press, February 1981, pp. 2-5.

3. Robert Ryan, et. al.,"Intel Local Network Architecture", Micro, IEEE Computer Society, Vol. 1, No. 4, November 1981, pp.26-41.

4. Andrew S. Tanenbaum, Computer Networks, Prentice-Hall, 1981.

5. Information Sciences Institute, University of Southern California, DOD Standard Transmission Protocol, Defense Advanced Research Projects Agency, January 1980.

6. Kenneth A. Inman, Jr. and Robert C. Marthouse, Jr., "Supply Point Logistics Integrated Communications Environment (SPLICE) Local Area Computer Network Design Issues for Communications", Masters Thesis, Naval Postgraduate School, June 1982.

7. V.G. Cerf and P.T. Kirstein, "Issues in Packet-Network Interconnection", Proceedings of the IEEE, November 1978, pp. 1386-1408.

8. Information Sciences Institute, University of Southern California, DOD Standard Internet Protocol, January 1980.

9. L. W. Davies, et. al., Computer Networks and Their Protocols", John Wiley α Sons, 1979.

10. J.D. Day, "Terminal Protocols", IEEE Transactions on Communications, Vol. Com-28, No. 4, April 1980, pp. 585-593.

11. J. B. Postel, "Internet Protocol Approaches", IEEE Transactions on Communications, Vol. Com-18, No. 4, April 1980, pp. 604-611.

12. Defense Data Network Program Plan, Defense Communications Agency, January 1982, Revised May 1982.

13. D. D. Clark, et. al., "An Introduction to Local Area Networks", Proceedings of the IEEE, Vol. 66, No. 11, November 1978, pp. 1497-1517.

14. Clifford Warner, "Connecting Local Networks to Long Haul Networks: Issues in Protocol Design", 5th Conference on Local Computer Networks, IEEE Computer Society, October 1980, pp. 71-76.

15. Fred J. Maryanski, "Backend Database Machines", ACM Computing Surveys, Vol. 12, No. 1, March 1980, pp. 3-25.

16. Jayantu Banerjee, David K. Hsiao and Krishnamurthi Kanna, "DBC-A Database Computer for Very Large Databases", IEEE Transactions on Computers, Vol. C-28, No. 6, June 1979, pp. 414-429.

17. Eugene Lowenthal, "Database Systems for Local Nets", Datamation, August 1982, pp. 96-106.

18. A. Rybczynski, "X.25 Interface and End-to-End Virtual Circuit Service Characteristics", Ieee Transactions on Communications, Vol. COM-28, No. 4, April 1980, pp. 500-510.

19. Vinton C. Cerf and Robert E. Kahn, "A Protocol for Packet Network Intercommunication", IEEE Transactions on Communications, Vol. COM-22, No. 5, May 1974, pp. 637-648.

20. Jonathan B. Postel et. al., "The ARPA Internet Protocol", Computer Networks, No. 5, 1981, pp. 261-171.

21. David R. Boggs, et. al., "Pup: An Internet Architecture", IEEE Transactions on Communications, Vol. COM-18, No. 4, April 1980, pp. 612-624.

22. Stuart Wecker, "DNA: The Digital Network Architecture", IEEE Transactions on Communications, Vol. COM-28, No. 4, April 1980, pp. 510-526.

23. J.D. Atkins, "Path Control: The Transport Network of SNA", IEEE Transactions on Communications, Vol. COM-28, No. 4, April 1980, pp. 527-538.

24. Hubert Zimmerman, "OSI Reference Model - The ISO Model of Architecture for Open Systems Interconnection", IEEE Transactions on Communications, Vol. COM-28, No. 4, April 1980, pp. 425-432.

25. Charles Bachman and Mike Canepa, "The Session Control Layer of an Open System Interconnection", Proceedings, Computer Communications Networks, COMPCON, September 1978, pp. 150-156.

26. Joseph N. Reinhart III and Ricardo Arana, "Database and Terminal Management Functional Design Specifications in Support of Stock Point Logistics Integrated Communication Environment (SPLICE)", Masters Thesis, Naval Postgraduate School, June 1982.

27. G.A. Champine, "Current Trends in Data Base Systems", Computer, IEEE Computer Society, Vol. 12, No. 5, May 1979, pp. 27-41.

## DISTRIBUTION LIST

| | No. copies |
|---|---|
| Prof. Charles Arnold<br>Computer Science Department<br>Naval Postgraduate School<br>Monterey, CA 93940 | 1 |
| Mr. James D. Atkins<br>IBM<br>Old Orchard Road<br>Armonk, NY 10504 | 1 |
| Prof. Dushan Badal<br>Computer Science Department<br>Naval Postgraduate School<br>Monterey, CA 93940 | 1 |
| Dr. D.R. Boggs<br>Palo Alto Research Center<br>Xerox Corporation<br>Palo Alto, CA 94304 | 1 |
| LCDR Steve Bristow<br>Navy Management System Support Office<br>NAS<br>Norfolk, VA 23464 | 1 |
| LCDR Ted Case<br>Fleet Material Support Office<br>Code 94L<br>Mechanicsburg, PA 17055 | 1 |
| Dr. Vinton G. Cerf<br>Defense Advanced Research Projects Agency<br>U.S. Department of Defense<br>Arlington, VA 22209 | 1 |
| Dr. David D. Clark<br>MIT Laboratory for Computer Science<br>Computer Systems and Communications Group<br>Cambridge, MA 02139 | 1 |
| Mr. John Day<br>Digital Technology Inc.<br>Champaign, IL 61801 | 1 |
| Dr. D.W. Davies<br>National Physical Laboratory<br>Teddington, England | 1 |

| | |
|---|---|
| Prof. Dan Dolk<br>Code 54Dk<br>Administrative Sciences Department<br>Naval Postgraduate School<br>Monterey, Ca 93940 | 1 |
| LCDR Dana Fuller<br>Commander, Naval Supply Systems Command<br>Code 0415A<br>Washington, D.C. 20376 | 1 |
| Dr. Harvey A. Freeman<br>Architecture Technology<br>P.O. Box 24344<br>Minneapolis, MN 55424 | 1 |
| Capt. Chuck Gibfried<br>COMNAVAIRPAC<br>Code 40,<br>NAS<br>San Diego, CA 92135 | 1 |
| LCDR John Hayes<br>Code 54Ht<br>Administrative Sciences Department<br>Naval Postgraduate School<br>Monterey, CA 93940 | 1 |
| Prof. Gordon Howell<br>Code 54Hv<br>Administrative Sciences Department<br>Naval Postgraduate School<br>Monterey, CA 93940 | 1 |
| Prof. David Hsaio<br>Code 52<br>Computer Science Department<br>Naval Postgraduate School<br>Monterey, CA 93940 | 1 |
| Ms. Tan Tahn Joo<br>Dy Head, Software Engineering Department<br>Information Engineering Centre<br>System Computer Organisation<br>Ministry of Defense<br>Minden Road, Singapore 1024 | 1 |
| Prof. Miles Kennedy<br>Weatherhead School of Management<br>Case Western Reserve University<br>Cleveland, Oh 44106 | 1 |

| | |
|---|---|
| Prof. Norm Lyons<br>Code 54Lb<br>Administrative Sciences Department<br>Naval Postgraduate School<br>Monterey, CA 93940 | 1 |
| Dr. R.M. Metcalfe<br>3 Com Corporation<br>3000 Sand Hill Road 1<br>Menlo Park, CA 94025 | 1 |
| Mr. Steve Oxman<br>c/o SHAPE Technical Center<br>United States Research and Development Coordinating Officer<br>The Hague, Netherlands | 1 |
| Dr. Kenneth T. Pogran<br>BBN Computer Corporation<br>10 Moulton Street<br>Cambridge, MA 01138 | 1 |
| Dr. Jonathan B. Postel<br>Information Sciences Institute<br>University of Southern California<br>Marina Del Rey, CA 90291 | 1 |
| Prof. George Rahe<br>Code 52Ra<br>Computer Science Department<br>Naval Postgraduate School<br>Monterey, CA 93940 | 1 |
| Mr. Antony Rybczynski<br>Computer Communications Group<br>Trans-Canada Telephone System<br>Ottawa, Ontario, Canada | 1 |
| Dr. Harry J. Saal<br>Nestar Systems, Inc.<br>2585 East Bayshore<br>Palo Alto, Ca 94303 | 1 |
| Prof. Norman F. Schneidewind<br>Code 54Ss<br>Administrative Sciences Department<br>Naval Postgraduate School<br>Monterey, CA 93940 | 30 |
| Dr. J.F. Shoch<br>Palo Alto Research Center<br>Xerox Corporation<br>Palo Alto, CA 94304 | 1 |

| | |
|---|---|
| Capt. William Sweeney<br>879 N. Lexington Street<br>Arlington, VA 22205 | 1 |
| Prof. Andrew S. Tanenbaum<br>Urije Universiteit<br>Amsterdam, The Netherlands | 1 |
| Dr. Kenneth Thurber<br>Architecuture Technology<br>P.O. Box 24344<br>Minneapolis, MN 55424 | 1 |
| CDR Bill Walton<br>Aviation Supply Office<br>700 Robbins Avenue<br>Philadelphia, PA 19111 | 1 |
| Mr. Clifford Warner<br>Naval Oceanographic Systems Center<br>201 Catalina Blvd.<br>San Diego, CA 92147 | 1 |
| Mr. Stuart Wecker<br>Corporate Research Group<br>Digital Equipment Corporation<br>Maynard, MA 01754 | 1 |
| Prof. Roger Weissinger-Baylon<br>Code 54Wr<br>Administrative Sciences Department<br>Naval Postgraduate School<br>Monterey, CA 93940 | 1 |
| Ms. Mary Willoughby<br>P.O. Box 94<br>Mendocino, CA 95460 | 1 |
| Mr. Gil Young<br>Aviation Supply Office<br>700 Robbins Avenue<br>Philadelphia, PA 19111 | 1 |
| Dr. Hubert Zimmermann<br>IRIA/Laboria<br>Rocquencourt, France | 1 |
| Administrative Sciences Department<br>Code 54<br>Naval Postgraduate School<br>Monterey, CA | 1 |

| | |
|---|---|
| Computer Center Library<br>Code 0141<br>Naval Postgraduate School<br>Monterey, CA 93940 | 1 |
| Computer Science Department<br>Code 52<br>Naval Postgraduate School<br>Monterey, CA 93940 | 1 |
| Defense Technical Information Center<br>Cameron Station<br>Alexandria, Va 23314 | 2 |
| Knox Library<br>Code 0142<br>Naval Postgraduate School<br>Monterey, CA 93940 | 4 |
| Office of Research Administration<br>Code 012A<br>Naval Postgraduate School<br>Monterey, CA 93940 | 1 |